TUGAS AKHIR – K141502

RANCANG BANGUN APLIKASI MUSICMOO DENGAN METODE MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL MOOD, GENRE RECOGNITION, DAN TEMPO ESTIMATION

JOHANES ANDRE R.
NRP 5113100021

Dosen Pembimbing
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dwi Sunaryono, S.Kom, M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – K141502

RANCANG BANGUN APLIKASI MUSICMOO DENGAN METODE MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL MOOD, GENRE RECOGNITION, DAN TEMPO ESTIMATION

JOHANES ANDRE R.
NRP 5113100021

Dosen Pembimbing
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dwi Sunaryono, S.Kom, M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – K141502

DESIGN AND IMPLEMENTATION OF MUSICMOO APPLICATION USING MIR (MUSIC INFORMATION RETRIEVAL) METHOD MOOD MODULE, GENRE RECOGNITION, AND TEMPO ESTIMATION

JOHANES ANDRE R.
NRP 5113100021

Supervisor
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dwi Sunaryono, S.Kom, M.Kom.

Department of Informatics
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI MUSICMOO DENGAN METODE MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL MOOD, GENRE RECOGNITION, DAN TEMPO ESTIMATION

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Manajemen Informasi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
JOHANES ANDRE R.
NRP: 5113 100 021

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
NIP: 19590803 198601 1 001 (Pembimbing 1)

Dwi Sunaryono, S.Kom., M.Kom.
NIP: 19720528 199702 1 001 (Pembimbing 2)

SURABAYA
JANUARI 2017

[Halaman ini sengaja dikosongkan]

RANCANG BANGUN APLIKASI MUSICMOO DENGAN METODE MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL MOOD, GENRE RECOGNITION, DAN TEMPO ESTIMATION

Nama Mahasiswa	: Johanes Andre R.
NRP	: 5113 100 021
Jurusan	: Teknik Informatika FTIf-ITS
Dosen Pembimbing 1	: Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dosen Pembimbing 2	: Dwi Sunaryono, S.Kom., M.Kom.

ABSTRAKSI

Saat ini, metode pemanggilan kembali informasi suatu musik atau yang sering disebut *Music Information Retrieval* (MIR) telah banyak diterapkan. Contohnya adalah pada suatu aplikasi Shazam ataupun Soundhound. Kedua aplikasi ini hanya mampu menangani sebatas deteksi judul lagu apa ketika diperdengarkan suatu musik. Untuk itu, Tugas Akhir ini adalah pengembangan lebih lanjut MIR yang lebih spesifik lagi, yaitu melakukan pemanggilan informasi lagu yang terkait kembali beserta detail lagu di antaranya adalah *mood*, *genre*, dan tempo lagu. Tujuan Tugas Akhir ini yaitu untuk meningkatkan pengetahuan akan informasi suatu lagu dan bisa juga untuk deteksi plagiarisme karya lagu seseorang.

Langkah pertama yang dilakukan adalah melakukan ekstraksi fitur audio berbasis MPEG-7 dengan *library* Java bernama *MPEG7AudioEnc.* Hasil dari ekstraksi fitur ini berupa metadata dengan XML yang di dalamnya terdapat fitur-fitur dalam bentuk angka-angka digital yang merepresentasikan karakteristik suatu sinyal. Kedua, melakukan pemilihan fitur yang dipakai dan diambil menggunakan *Xquery* untuk dilakukan proses sesuai dengan masing-masing modul. Pemrosesan pada fitur-fitur yang dipilih adalah melakukan *Discrete Wavelet Transform* (DWT)

beserta level dekomposisi terbaik menggunakan *library pywt*. Setelah fitur-fitur dilakukan DWT, maka dilakukan penggabungan fitur ke suatu *list* beserta penyamaan panjang fitur untuk proses klasifikasi. Tahap terakhir adalah melakukan klasifikasi menggunakan *Support Vector Machine* (SVM). Tahapan proses SVM terdiri dari 2 tahap yaitu tahap *training* dan prediksi. Tahap *training* adalah melakukan pembelajaran karakteristik sinyal sesuai dengan label yang dimaksud, sedangkan tahap prediksi adalah memprediksi data baru yang belum diketahui dan akan memberikan suatu penilaian sesuai dengan tahap *training*. Hasilnya adalah detail informasi *mood*, *genre*, *tempo* pada suatu lagu berdasarkan karakteristik sinyal.

Kata Kunci: Analisis Audio, MIR, MPEG-7, SVM.

DESIGN AND IMPLEMENTATION OF MUSICMOO APPLICATION USING MIR (MUSIC INFORMATION RETRIEVAL) METHOD MOOD MODULE, GENRE RECOGNITION, AND TEMPO ESTIMATION

Student Name	:	Johanes Andre R.
Student ID	:	5113 100 021
Major	:	Informatics Department FTIf-ITS
Advisor 1	:	Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Advisor 2	:	Dwi Sunaryono, S.Kom, M.Kom.

ABSTRACT

Currently, the method of recalling information a music or often called Music Information Retrieval (MIR) *has been widely applied. For example are application Shazam or Soundhound. Both of these applications can only handle detection of music titles when played some music. Therefore, this Final Project is a further development of more specific* MIR, *ie to call back the track information associated with specific songs among which are the mood, genre, and tempo of the music. The purpose of this Final Project is to improve the knowledge and information of a music and track music for plagiarism detection.*

The first step taken is extracting audio features based on MPEG-7 *with a* Java *library named MPEG7AudioEnc. The results of this feature extraction are metadata* XML *which are includes digital numbers that represent the characteristics of a signal. Second, do a selection of used features and retrieved using XQuery to do the process in accordance with each module. Processing on features selected is doing Discrete Wavelet Transform* (DWT) *and their the best decomposition level using* Python *library named pywt. After doing* DWT *for selected features, then do a merger of features on a list along with a feature length equation for the classification process. The last step is a classification using Support Vector Machine* (SVM). SVM *stage of the process*

consists of two phases: training and prediction. The training phase is doing the learning signal characteristics in accordance with the corresponding label, while the prediction phase is predicting new data is not yet known and will provide an assessment in accordance with the training phase. The results are a detail of information mood, genre, tempo on a music based on the signal characteristics.

Keywords: Audio Analysis, MIR, MPEG-7, SVM.

KATA PENGANTAR

Puji syukur kepada Tuhan Yesus Kristus atas segala kasih karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul:

RANCANG BANGUN APLIKASI MUSICMOO DENGAN METODE MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL MOOD, GENRE RECOGNITION, DAN TEMPO ESTIMATION

Melalui lembar ini, penulis ingin menyampaikan ucapan terimakasih dan penghormatan yang sebesar-besarnya kepada:

1. Bapak, Ibu, kakak dan keluarga besar yang selalu memberikan dukungan penuh untuk menyelesaikan Tugas Akhir ini.
2. Prof. Drs. Ec. Ir. Riyanarto Sarno, M.sc., Ph.D., selaku dosen pembimbing 1 yang telah banyak membantu, membimbing, bahkan memotivasi penulis untuk menyelesaikan Tugas Akhir ini.
3. Bapak Dwi Sunaryo selaku dosen pembimbing 2 yang telah banyak memberikan semangat, motivasi, serta arahan kepada penulis dalam menyelesaikan Tugas Akhir ini.
4. Bapak Dedy Rahman Wijaya, selaku mahasiswa S-3 sekaligus anak bimbing Prof. Riyanarto yang telah banyak membimbing dan memberikan arahan kepada penulis dalam menyelesaikan Tugas Akhir ini.
5. Faris Ponighzwa, selaku tim yang membantu dalam pengerjaan Tugas Akhir ini.
6. Lidya Cintyabudi, selaku rekan sekaligus kekasih yang selalu mendukung dan menyemangati penulis bahkan di saat hampir menyerah.
7. David, Andrew, Albert, Alvin, Andre, Freddy, Arianto, Billy, Romario, dan Yosua selaku sahabat-sahabat karib

yang banyak membantu penulis selama perkuliahan di Jurusan Teknik Informatika ITS.

8. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS lainnya yang telah banyak menyampaikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
9. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu per satu.

Penulis menyadari masih terdapat banyak kekurangan baik dalam pelaksanaan Tugas Akhir maupun penyusunan buku laporan ini, namun penulis berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi. Penulis mengharapkan kritik dan saran yang membangun untuk kesempurnaan penulisan buku Tugas Akhir ini.

Surabaya, Desember 2016

Johanes Andre R

DAFTAR ISI

LEMBAR PENGESAHAN vii
ABSTRAKSI ix
ABSTRACT xi
KATA PENGANTAR xiii
DAFTAR ISI xv
DAFTAR GAMBAR xix
DAFTAR TABEL xxi
DAFTAR KODE SUMBER xxiii
DAFTAR PERSAMAAN xxv
BAB I PENDAHULUAN 1
1.1. Latar Belakang 1
1.2. Tujuan 2
1.3. Rumusan Permasalahan 2
1.4. Batasan Permasalahan 3
1.5. Metodologi 3
1.6. Sistematika Penulisan 5
BAB II DASAR TEORI 7
2.1. Music Information Retrieval 7
2.2. Aplikasi yang Sudah Ada 7
2.2.1. SoundHoud 7
2.2.2. Shazam 8
2.3. WAV 9
2.4. MPEG-7 9
2.5. MPEG-7 Low Level Audio Descriptors 11
2.5.1. Audio Power 13
2.5.2. Audio Waveform 13
2.5.3. Temporal Centroid 13
2.5.4. Log Attack Time 14
2.5.5. Audio Spectrum Centroid 14

2.5.6. Audio Harmonicity 16
2.5.7. Harmonic Spectral Centroid 16
2.5.8. Harmonic Spectral Deviation 17
2.5.9. Harmonic Spectral Spread 18
2.5.10. Harmonic Spectral Variation 18
2.5.11. Audio Spectrum Spread Type 19
2.5.12. Audio Spectrum Projection 20
2.5.13. Audio Spectrum Basis Type 20
2.5.14. Audio Spectrum Flatness Type 20
2.5.15. Audio Spectrum Envelope Type 21
2.5.16. Audio Fundamental Frequency 22
2.5.17. Audio Signature 22
2.5.18. Sound Model 23
2.6. Fast Fourier Transform 23
2.7. Discrete Wavelet Transform 23
2.8. Xquery 25
2.9. Valence 25
2.10. Arousal 25
2.11. Dataset 25
2.12. Mood 26
2.13. Genre 26
2.14. Tempo 27
2.15. Russel's Circumplex models 28
2.16. Support Vector Machine 28
BAB III ANALISIS DAN PERANCANGAN SISTEM 31
3.1. Analisis 31
3.1.1. Kontribusi Penelitian 31
3.1.2. Analisis Permasalahan 32
3.1.3. Analisis Kebutuhan 32

3.1.4. Deskripsi Umum Sistem 33
3.1.5. Kasus Penggunaan 43
3.2. Perancangan Sistem 44
3.2.1. Perancangan Basis Data 45
3.2.2. Perancangan Antar Muka 47
3.2.3. Perancangan Alur Proses Penggunaan Aplikasi .. 49
4. BAB IV IMPLEMENTASI 51
4.1. Lingkungan Implementasi 51
4.1.1. Lingkungan Implementasi Perangkat Keras 51
4.1.2. Lingkungan Implementasi Perangkat Lunak 51
4.2. Implementasi Server 51
4.2.1. Implementasi Konfigurasi Server dengan Flask .. 51
4.2.2. Implementasi Mengubah Sinyal dari Time Domain menjadi Frekuensi Domain 52
4.2.3. Implementasi Mencari Nilai Dekomposisi Terbaik. 53
4.2.4. Implementasi Discrete Wavelet Transform 54
4.2.1. Implementasi Reduksi Data 54
4.2.2. Implementasi Gabung Fitur 55
4.2.3. Implementasi SVM 56
4.3. Implementasi Perangkat Bergerak 58
4.3.1. Implementasi Antar Muka 59
4.3.2. Implementasi Perekaman Audio 60
4.3.3. Implementasi Memberhentikan Perekaman Audio 60
4.3.4. Implementasi Penyimpanan Hasil Rekaman Audio 61
4.3.5. Implementasi Upload Lagu Ke Server 62

4.4. Implementasi Ekstraktor MPEG-7 62
4.4.1. Implementasi Mengimpor Library yang Digunakan 62
4.4.2. Implementasi Ekstraktor basis MPEG-7 62
4.5. Implementasi XQuery untuk Mengambil Fitur Audio 64
4.5.1. Mengambil Nilai Audio Power 64
4.5.2. Mengambil Nilai Audio Harmonicity 65
4.5.3. Mengambil Nilai Audio Spectrum Centroid 66
4.5.4. Mengambil Nilai Audio Spectrum Spread 67
4.5.5. Mengambil Nilai Audio Spectrum Flatness 68
BAB V PENGUJIAN DAN EVALUASI 71
5.1. Lingkungan Pengujian 71
5.2. Skenario Pengujian 71
5.2.1. Pengujian Fungsionalitas 73
5.3. Akurasi Pengujian Fungsionalitas 77
5.3.1. Hasil Percobaan Modul Mood 79
5.3.2. Hasil Percobaan Modul Tempo 80
5.3.3. Hasil Percobaan Modul Genre 81
5.4. Evaluasi Pengujian 81
5.4.1. Evaluasi Pengujian Fungsionalitas 81
BAB VI KESIMPULAN DAN SARAN 83
6.1. Kesimpulan 83
6.2. Saran 83
DAFTAR PUSTAKA 85
LAMPIRAN A 89
LAMPIRAN B 95
BIODATA PENULIS 99

DAFTAR GAMBAR

Gambar 2.1 Aplikasi SoundHound 8
Gambar 2.2 Aplikasi Shazam 8
Gambar 2.3 Tahapan Ekstraksi Fitur MPEG-7 9
Gambar 2.4 Contoh Isi Data Dari Metadata MPEG-7 10
Gambar 2.5 Contoh Plot Angka Digital Menjadi Sinyal 10
Gambar 2.6 Syntax Metadata MPEG-7 LLADs 11
Gambar 2.7 Hirarki Kelas Pada MPEG-7 LLADs 12
Gambar 2.8 Tahapan Proses Xquery 25
Gambar 2.9 Tahapan Mendapat BPM Suatu Lagu 27
Gambar 2.10 Bagan Russel's Circumplex 28
Gambar 2.11 Proses Klasifikasi SVM 29
Gambar 3.1 Deskripsi Umum Sistem 33
Gambar 3.2 Ekstaksi Sinyal Wav 35
Gambar 3.3 Tahapan Ekstraksi Fitur Pada Server Java 36
Gambar 3.4 Tahapan Processing Pada Server Python 37
Gambar 3.5 Tahapan Melakukan DWT 38
Gambar 3.6 Contoh Sinyal Setelah Terwavelet 39
Gambar 3.7 Gabungan Fitur AP dan AH 41
Gambar 3.8 Gabungan Fitur ASC, ASS, dan ASF 41
Gambar 3.9 Diagram Kasus Penggunaan 43
Gambar 3.10 Contoh Isi Data Pada Tabel Modul Mood 46
Gambar 3.11 Contoh Isi Data Pada Tabel Modul Genre 46
Gambar 3.12 Contoh Isi Data Pada Tabel Modul Tempo 47
Gambar 3.13 Perancangan Tampilan Awal Aplikasi 47
Gambar 3.14 Perancangan Tampilan Menu Dalam Button 48
Gambar 3.15 Perancangan Tampilan Hasil 49
Gambar 3.16 Workflow Alur Penggunaan Aplikasi 49
Gambar 4.1 Tampilan Awal Antarmuka Pada Perangkat Bergerak 59
Gambar 4.2 Tampilan Menu Di Dalam Tombol Pada Perangkat Bergerak 59
Gambar 5.1 Contoh Tampilan Output Pada Aplikasi Perangkat Bergerak 82

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

Tabel 2.1 Penentuan Decompose Level Wavelet 24
Tabel 2.2 Perbedaan Antara Emosi dan Mood 26
Tabel 2.3 Pembagian Estimasi Tempo 27
Tabel 3.1 Kebutuhan Fungsional Sistem 33
Tabel 3.2 Panjang Minimal Tiap Fitur 40
Tabel 3.3 Perincian Data Training 42
Tabel 3.4 Daftar Kode Kasus Penggunaan 43
Tabel 3.5 Perancangan Tabel Modul Mood 45
Tabel 3.6 Perancangan Tabel Modul Genre 45
Tabel 3.7 Perancangan Tabel Modul Tempo 46
Tabel 5.1 Skenario Pengujian 73
Tabel 5.2 Tabel Pengujian Modul Mood 73
Tabel 5.3 Tabel Pengujian Modul Genre 74
Tabel 5.4 Tabel Pengujian Modul Mood 76
Tabel 5.6 Pemilihan Tiap Fitur 77
Tabel 5.7 Rincian Jumlah Data Testing 79
Tabel 5.8 Hasil Percobaan Modul Mood 80
Tabel 5.9 Hasil Percobaan Modul Tempo 80
Tabel 5.10 Hasil Percobaan Genre 81
Tabel 5.5 Rangkuman Hasil Pengujian 82
Tabel A.1 Hasil Percobaan Modul Mood 95
Tabel A.2 Hasil Percobaan Modul Genre 96
Tabel A.3 Hasil Percobaan Modul Tempo 97

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

Kode Sumber 4.1 Konfigurasi Server Flask 52
Kode Sumber 4.2 Mengubah Sinyal dari Time Domain Menjadi Frequency Domain 53
Kode Sumber 4.3 Mencari Nilai Dekomposisi Terbaik 54
Kode Sumber 4.4 Melakukan Wavelet 54
Kode Sumber 4.5 Implementasi Reduksi Data 55
Kode Sumber 4.6 Implementasi Gabung Fitur 56
Kode Sumber 4.7 SVM Pada Modul Mood 57
Kode Sumber 4.8 SVM Pada Modul Genre 57
Kode Sumber 4.9 SVM Pada Modul Tempo 58
Kode Sumber 4.10 Implementasi Perekaman Audio 60
Kode Sumber 4.11 Memberhentikan Perekaman Audio 61
Kode Sumber 4.12 Penyimpanan Hasil Rekaman Audio 61
Kode Sumber 4.13 Mengimpor Library yang Digunakan 62
Kode Sumber 4.14 Implementasi Ektraktor basis MPEG-7 64
Kode Sumber 4.15 Mengambil Nilai Audio Power 65
Kode Sumber 4.16 Mengambil Nilai Audio Harmonicity 66
Kode Sumber 4.17 Mengambil Nilai Audio SpectrumCentroid 67
Kode Sumber 4.18 Mengambil Nilai Audio Spectrum Spread 68
Kode Sumber 4.19 Mengambil Nilai Audio Spectrum Flatness 69
Kode Sumber A.1 Upload Lagu ke Server 93

[Halaman ini sengaja dikosongkan]

DAFTAR PERSAMAAN

Persamaan 2.1 13
Persamaan 2.2 14
Persamaan 2.3 14
Persamaan 2.4 14
Persamaan 2.5 15
Persamaan 2.6 15
Persamaan 2.7 15
Persamaan 2.8 15
Persamaan 2.9 15
Persamaan 2.10 16
Persamaan 2.11 16
Persamaan 2.12 17
Persamaan 2.13 17
Persamaan 2.14 17
Persamaan 2.15 18
Persamaan 2.16 18
Persamaan 2.17 19
Persamaan 2.18 19
Persamaan 2.19 19
Persamaan 2.20 20
Persamaan 2.21 20
Persamaan 2.22 21
Persamaan 2.23 21
Persamaan 2.24 21
Persamaan 2.25 21
Persamaan 2.26 21
Persamaan 2.27 22
Persamaan 2.28 22
Persamaan 2.29 23
Persamaan 2.30 24
Persamaan 2.31 24
Persamaan 5.1 78

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1. Latar Belakang

Dewasa ini, industri media sudah berkembang sangat pesat, khususnya pada lagu. Lagu dapat ditemukan oleh masyarakat secara mudah. Mulai dari radio, *Compact Disc* (CD), internet, dan sumber-sumber lainnya.

Sebagai hasil dari ledakan terbaru dalam media tersebut, muncul juga suatu kebutuhan pokok untuk kalangan masyarakat agar bisa mengetahui informasi lagu yang lebih lengkap pada suatu lagu. Menjawab kebutuhan masyarakat tersebut, ditemukanlah metode *Music Information Retrieval* atau yang disingkat menjadi MIR. MIR adalah metode pemanggilan informasi suatu musik agar dapat memberikan informasi lagu yang kompleks[1].

Pada kali ini, Tugas Akhir ini akan menjawab implementasi kebutuhan masyarakat mengenai MIR. Penelitian yang dilakukan terkait dengan analisis dan implementasi MIR pada modul *mood*, *genre recognition*,dan *tempo estimation*. Tujuan dari Tugas Akhir ini adalah memperkaya detail informasi suatu lagu dengan memberikan informasi mengenai *mood*, *genre*, dan tempo sebuah lagu.

Konsep yang digunakan pada Tugas Akhir ini adalah menggunakan ekstraksi fitur berbasis MPEG-7 yang sudah menjadi standar dalam konten multimedia berdasarkan ISO/IEC 15938 [2]. Hasil ekstraksi fitur ini berupa metadata dalam format XML. Untuk mengimplementasikan ekstraksi fitur berbasis MPEG-7 ini menggunakan *library* Java bernama *MPEG7AudioEnc* yang bersifat *open source*. Di dalam file metadata XML tersebut berisi terdapat fitur-fitur dalam bentuk

angka-angka digital yang merepresentasikan karakteristik suatu sinyal. Setelah mendapatkan fitur ini, tahap berikutnya adalah melakukan pemilihan fitur yang dipakai dan diambil menggunakan *Xquery*. Fitur yang dipilih ini nantinya digunakan untuk pemrosesan sesuai dengan masing-masing modul. Pemrosesan yang dimaksud adalah melakukan *Discrete Wavelet Transform* (DWT) beserta level dekomposisi terbaik menggunakan *library pywt*. Kemudian melakukan penggabungan fitur pada suatu list beserta penyamaan panjang fitur untuk proses klasifikasi. Tahap terakhir adalah melakukan klasifikasi dengan menggunakan *Support Vector Machine* (SVM). Terdiri dari 2 tahap yaitu tahap *training* dan predisi. Tahap *training* adalah melakukan pembelajaran karakteristik sinyal sesuai dengan label yang dimaksud, sedangkan tahap prediksi adalah memprediksi data baru yang belum diketahui dan akan memberikan suatu penilaian sesuai dengan tahap *training.* Hasilnya adalah detail informasi *mood, genre, tempo* pada suatu lagu berdasarkan karakteristik sinyal.

1.2. Tujuan

Tujuan dari pembuatan tugas akhir ini adalah:

1. Mengetahui fitur-fitur lagu apakah yang dimiliki suatu lagu berbasis MPEG-7.
2. Mengetahui fitur lagu apakah yang mempengaruhi dalam menentukan *mood* suatu lagu.
3. Mengetahui fitur lagu apakah yang mempengaruhi dalam menentukan *genre* sebuah lagu.
4. Mengetahui fitur lagu apakah yang mempengaruhi dalam menentukan tempo sebuah lagu.

1.3. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini antara lain:

1. Bagaimana fitur musik yang dihasilkan lewat ekstraksi fitur audio berbasis MPEG -7?
2. Bagaimana cara identifikasi *mood* pada suatu musik?

3. Bagaimana cara identifikasi *genre* pada suatu musik?
4. Bagaimana cara identifikasi *tempo* pada suatu musik?
5. Bagaimana implementasi MIR menggunakan modul *mood, genre recognition*,dan *tempo estimation*?

1.4. Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, antara lain:

1. Hasil dari tugas akhir ini adalah sebuah aplikasi MusicMoo dengan modul *Mood, Genre Recognition*, dan *Tempo Estimation.*
2. Audio yang digunakan berekstensi .wav.
3. *Mood* pada suatu lagu hanya sebatas *angry*, *happy*, *relaxed*, dan *sad*.
4. *Genre* pada suatu lagu hanya sebatas *classic*, *electronic*, *jazz*, dan *rock*.
5. Tempo pada suatu lagu hanya sebatas *fast*, *medium*, dan *slow*.
6. Pada saat pengujian, teknik perekaman suatu lagu bisa berubah-ubah karena dipengaruhi oleh *speaker* sumber suara maupun sumber perekam.

1.5. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

a. Penyusunan proposal tugas akhir

Proposal tugas akhir ini berisi latar belakang pembuatan tugas akhir, rumusan masalah, batasan masalah, tujuan pembuatan, manfaat, metodologi hingga jadwal kegiatan pembuatan tugas akhir. Selain itu proposal tugas akhir ini memberikan ringkasan dari tugas akhir. Proposal tugas akhir juga berisi tinjauan pustaka yang digunakan sebagai referensi pembuatan tugas akhir ini.

b. Studi literatur

Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan program yaitu mengenai MPEG-7, MPEG-7 *Low Level Audio Descriptors*, *Xquery*, dan *Discrete Wavelet Transform* (DWT).

c. Analisis dan desain perangkat lunak

Tahap ini meliputi perancangan sistem berdasarkan studi literatur dan pembelajaran konsep teknologi dari perangkat lunak yang ada. Tahap ini mendefinisikan alur dari implementasi. Langkah-langkah yang dikerjakan juga didefinisikan pada tahap ini. Pada tahapan ini dibuat *prototype* sistem, yang merupakan rancangan dasar dari sistem yang akan dibuat. Serta dilakukan desain suatu sistem dan desain proses-proses yang ada.

d. Implementasi perangkat lunak

Implementasi perangkat lunak ini dibangun dengan sistem perangkat bergerak yang berbasis bahasa pemrograman *Java* dan *database* menggunakan MySQL. Sedangkan untuk *pre-processing* menggunakan bahasa Python.

e. Pengujian dan evaluasi

Pada tahapan ini dilakukan uji coba terhadap perangkat lunak yang telah dibuat. Pengujian yang dimaksud adalah pengujian fungsionalitas aplikasi yang dibangun. Pengujian dan evaluasi akan dilakukan dengan melihat kesesuaian dengan perencanaan. Tahap ini dimaksudkan juga untuk mengevaluasi jalannya aplikasi, mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

f. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat

lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
    a. Latar Belakang
    b. Rumusan Masalah
    c. Batasan Tugas Akhir
    d. Tujuan
    e. Metodologi
    f. Sistematika Penulisan
2. Tinjauan Pustaka
3. Desain dan Implementasi
4. Pengujian dan Evaluasi
5. Kesimpulan dan Saran
6. Daftar Pustaka

1.6. Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini. Teori yang terkait adalah ekstraksi fitur berbasis MPEG-7,

Discrete Wavelet Transform (DWT), *mood*, *genre*, dan tempo.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai perancangan perangkat lunak. Perancangan perangkat lunak meliputi perancangan data, arsitektur, proses dan perancangan antarmuka aplikasi.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan dan implementasi fitur-fitur penunjang aplikasi.

Bab V Pengujian dan Evaluasi

Membahas tentang lingkungan pengujian, skenario pengujian, dan evaluasi pengujian setelah aplikasi selesai dikembangkan.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

Lampiran

Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini.

BAB II
DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan Tugas Akhir.

2.1. *Music Information Retrieval*

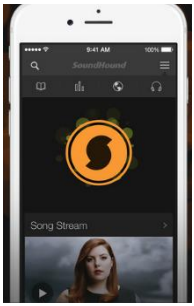
Music Information Retrieval (MIR) sesuai dengan definisi Downie adalah usaha penelitian yang berusaha untuk mengembangkan inovatif musik berbasis konten, mencari skema, *interface* baru untuk membuat toko besar di dunia musik [1]. MIR bisa diartikan sebagai pemanggilan informasi suatu musik agar dapat memberikan informasi lagu yang kompleks.

Sudah banyak penelitian terkait mengenai audio yang mengimplementasikan MIR. Misalnya adalah *query by humming* yaitu memanggil lagu dengan bergumam [3], *cover song identification* yaitu mendeteksi lagu yang dinyanyikan kembali [4], *pathological voice detection* yaitu deteksi suara patologis [5], dan lain-lain. Berbagai macam metode ekstraksi fitur pada audio pun beraneka ragam. Namun pada pengerjaan Tugas Akhir ini, metode yang digunakan adalah ekstrasi fitur dengan basis MPEG-7 dan berfokus pada klasikasi *mood*, *genre*, dan tempo suatu lagu yang akan dijelaskan di Bab III.

2.2. Aplikasi yang Sudah Ada

2.2.1. SoundHoud

SoundHound adalah aplikasi yang memungkinkan pengguna untuk mengidentifikasi judul lagu dengan dengungan, siulan, dan musik yang sedang dimainkan [6]. Contoh aplikasi SoundHound diperlihatkan pada Gambar 2.1.



Gambar 2.1 Aplikasi SoundHound

2.2.2. Shazam

Shazam adalah aplikasi yang membantu kita mencari tahu sebuah judul lagu yang sedang didengungkan [7]. Contoh aplikasi Shazam diperlihatkan pada Gambar 2.2.

Gambar 2.2 Aplikasi Shazam

Kedua aplikasi ini mampu mendeteksi suatu lagu berdasarkan karakteristik suara, namun masih hanya sebatas judul lagu saja. Oleh karena itu, Tugas Akhir ini dibangun untuk memberikan informasi suatu lagu yang lebih detail sehingga suatu lagu akan lebih kaya informasinya.

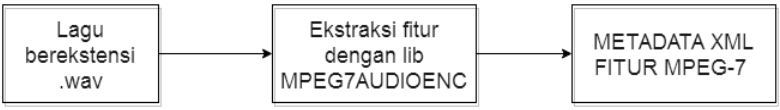
2.3. *WAV*

WAV adalah singkatan dari istilah dalam bahasa Inggris *Waveform Audio Format* merupakan standar format berkas audio yang dikembangkan oleh Microsoft dan IBM. Format WAV banyak digunakan oleh *handphone*, sehingga popularitas bisa menyamai file MP3.

2.4. MPEG-7

MPEG-7 adalah deskripsi standar konten multimedia dalam ISO/IEC 15938 [8]. Dari deskripsi yang terkait, konten tersebut dapat memungkinkan pencarian cepat dan effisien.

Ekstraksi fitur yang dilakukan ini menggunakan *library* Java yang bernama *MPEG7AudioEnc* yang bisa didapatkan pada *sourceforge*. *Input*nya adalah sebuah lagu bereksensi wav, dilakukan ekstraksi fitur. Hasil ekstraksi fitur pada MPEG-7 ini berupa metadata dengan format XML. Gambar 2.3 adalah tahapan untuk melakukan ekstraksi fitur berbasis MPEG-7.



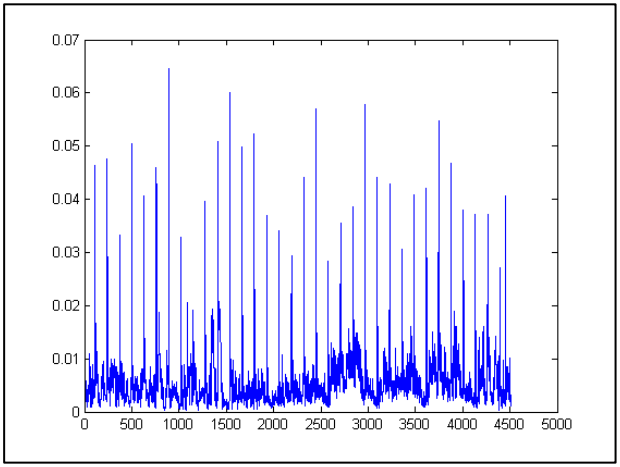
Gambar 2.3 Tahapan Ekstraksi Fitur MPEG-7

Hasil dari ekstraksi audio berbasis MPEG-7 ini adalah sebuah metadata XML yang berisi fitur-fitur berupa karakteristik sinyal. Sinyal yang dihasilkan berupa nilai digital yang panjang. Gambar 2.4 adalah contoh isi data dari XML yang dihasilkan oleh ekstraksi MPEG-7 yang merupakan contoh isi data pada fitur *Audio Power*. Data sinyal berupa angka-angka digital dalam

```
</Raw></SeriesOfScalar></UpperLimitOfHarmonicity></AudioDescriptor><AudioDescriptor xsi:type="AudioPowerType"><SeriesOfScalar hopSize="PT10N1000F"
totalNumOfSamples="4508"><Raw>0.0 0.0 6.757896E-11 4.5815136E-6 3.2904052E-5 3.4488403E-5 4.4055378E-5 3.1270883E-5 2.878371E-5 2.8276318E-5 3.7766105E-5
2.7014117E-5 3.9766284E-5 1.9448198E-5 1.9136802E-5 2.8626175E-5 2.011027E-5 2.2653594E-5 2.3187336E-5 1.2699299E-5 1.2039114E-5 4.4056733E-6 4.633146E-6
6.406585E-6 5.7510647E-6 9.815552E-6 3.875755E-6 8.5812735E-6 8.953873E-6 6.0477428E-6 3.953473E-6 7.506895E-6 3.8665407E-6 3.5911123E-6 5.778706E-5
0.0021328935 0.0069570723 0.007049585 0.0049764947 0.004210572 0.004178834 0.0063359877 0.0057332693 0.006155226 0.005608983 0.0062818555 0.0067735547
0.005084108 0.0034875455 0.0029903075 0.003428247 0.0045787916 0.0052722786 0.0051788352 0.0059483293 0.006681054 0.0067263194 0.0074646464 0.007397632
0.006502689 0.0057625044 0.0050914213 0.0050003855 0.0050545046 0.004969644 0.0048284847 0.0049023163 0.0048581953 0.004863815 0.004941797 0.004691671
0.004291982 0.0040212725 0.0036185256 0.0033670217 0.0031339915 0.0029430178 0.0027565483 0.0025371588 0.0023264084 0.0022042254 0.0019952073 0.0017434198
0.0015811616 0.0013872512 0.0012030818 0.0010563035 9.401313E-4 7.6175696E-4 6.170252E-4 5.196928E-4 4.6939E-4 3.9954833E-4 3.6172E-4 2.8449888E-4
2.4646585E-4 4.0583857E-4 0.012295584 0.07410384 0.08676415 0.13573341 0.14792123 0.11150838 0.08283244 0.06464404 0.043826148 0.024384199 0.013268818
0.009456868 0.00769835 0.007057071 0.006552733 0.0061454494 0.005219651 0.0047519 0.0052466546 0.004680657 0.0038680483 0.0032779046 0.0028865018
0.0022983756 0.0016499056 0.001051384 7.830468E-4 6.7472935E-4 5.615104E-4 6.525789E-4 7.332019E-4 7.239527E-4 7.35154E-4 5.01073E-4 3.6393775E-4
2.9672094E-4 2.5675393E-4 2.7132736E-4 0.006173165 0.017849037 0.014949953 0.021423021 0.009746517 0.0032799388 0.0031291589 8.2930323E-4 0.0010431454
9.0168265E-4 2.7827543E-4 3.432617E-4 2.606257E-4 7.186526E-5 9.2647555E-5 2.5626444E-4 1.9704555E-4 5.9139533E-5 1.5166146E-4 3.531694E-5 4.8983497E-5
8.3896826E-5 1.0435635E-4 2.262883E-4 0.0010240877 0.025180124 0.03927545 0.053139053 0.048805833 0.033853408 0.027622737 0.016899662 0.009504017 0.008066149
0.004858067 0.004493008 0.0038001598 0.0035502855 0.0034006871 0.0039526755 0.0031681082 0.00212173 0.0019057256 0.0013088818 0.001032002 0.0010910254
8.396533E-4 7.816088E-4 7.9826347E-4 6.024102E-4 4.7590592E-4 5.306261E-4 4.6406727E-4 4.934379E-4 4.3738922E-4 3.8011707E-4 3.1350303E-4 2.931304E-4
1.9978198E-4 2.1074855E-4 1.7610811E-4 4.2199838E-4 0.0077912044 0.017662242 0.012543642 0.017596325 0.005940225 0.0027218086 0.0021900716 5.661888E-4
```


Gambar 2.4 Contoh Isi Data Dari Metadata MPEG-7

jumlah yang sangat banyak. Ketika angka-angka ini di-*plot* maka terlihatlah fitur ini merupakan data dalam bentuk sinyal. Gambar 2.5 merupakan contoh *plotting* angka digital dari isi data pada Gambar 2.4.



Gambar 2.5 Contoh Plot Angka Digital Menjadi Sinyal

Ekstraksi fitur berbasis MPEG-7 ini mempunyai keunggulan yaitu:

1. Merupakan standar industri ISO/IEC 15938.

2. Data yang dihasilkan berupa metadata, sehingga lebih efisien.
3. Kaya akan fitur yang bisa digunakan untuk menggambarkan konten suatu multimedia. Sehingga fitur ini bisa digunakan untuk berbagai macam MIR.

2.5. MPEG-7 Low Level Audio Descriptors

MPEG-7 *Low Level Descriptors (*LLADs) adalah deskriptor yang disimpan menggambarkan variasi sifat frekuensi audio dari waktu ke waktu. Fungsinya adalah untuk mengidentifikasi identik suatu lagu apakah sama atau berbeda.

Pada metadata XML yang dihasilkan memiliki *syntax* seperti pada Gambar 2.6 di mana *name*=”*AudioLLDScalarType*” adalah nama fitur yang dihasilkan. Secara garis besar, kelas hirarki pada XML metadata yang dihasilkan oleh MPEG-7 adalah seperti pada Gambar 2.7 [9].

```
<!-- ################################################################################ -->
<!-- Definition of AudioLLDScalar datatype                                          -->
<!-- ################################################################################ -->
<complexType name="AudioLLDScalarType" abstract="true">
  <complexContent>
    <extension base="mpeg7:AudioDType">
      <choice>
        <element name="Scalar" type="float"/>
        <element name="SeriesOfScalar" minOccurs="1" maxOccurs="unbounded">
          <complexType>
            <complexContent>
              <extension base="mpeg7:SeriesOfScalarType">
                <attribute name="hopSize" type="mpeg7:mediaDurationType"
                           default="PT10N1000F"/>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </choice>
      <attribute name="confidence" type="mpeg7:zeroToOneType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

Gambar 2.6 Syntax Metadata MPEG-7 LLADs

Gambar 2.7 Hirarki Kelas Pada MPEG-7 LLADs

Berikut adalah penjelasan setiap *Low Level Audio Descriptors* yang dihasilkan [8][2]. Namun pada Tugas Akhir ini yang digunakan hanyalah *Audio Power*, *Audio Harmonicity*, *Audio Specrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*. Untuk penjelasan akan dipaparkan pada Bab III.

2.5.1. *Audio Power*

Audio Power (AP) adalah fitur yang menggambarkan temporal daya sesaat dari sinyal audio. Koefisiennya dari kuadrat rata-rata nilai gelombang *s(n)* dalam *non-overlapping frame.* Tujuannya adalah untuk membandingkan suatu label sinyal. Persamaan 2.1 adalah cara mendapatkan nilai AP.

$$AP(l) = \frac{1}{N_{hop}} \sum_{n=0}^{N_{hop}-1} \left|s(n + lN_{hop})\right|^2 \ (0 \ \leq l \leq L-1) \tag{2.1}$$

di mana:

L = total jumlah frame waktu.
s(n) = rata-rata *square waveform.*
l = indeks frame.
N_{hop} = sampel antara *successive non-overlapping.*

2.5.2. *Audio Waveform*

Audio Waveform adalah daya temporal yang diberikan oleh nilai *hopSize*, yang terdiri dalam menggambarkan setiap frame garis vertikal dari *min range* ke *max range*. Tujuannya untuk menampilkan ringkasan dari file audio. Dengan demikian, fitur ini dapat digunakan untuk perbandingan kecepatan gelombang.

2.5.3. Temporal Centroid

Temporal Centroid (TC) mendefinisikan sebagai rata-rata sampling sinyal *envelope.* Persamaan 2.2 adalah cara mendapatkan nilai TC.

$$TC = \frac{N_{hop}}{F_s} \frac{\sum_{l=0}^{L-1}(lEnv(l))}{\sum_{l=0}^{L-1} Env(l)} \tag{2.2}$$

di mana: *Env*(*l*) diperoleh dari Persamaan 2.3.

$$Env(l) = \sqrt{\frac{1}{N_w}\sum_{n=0}^{N_{w-1}} s^2\left(lN_{hop}+n\right)} (0 \le l \le L-1) \tag{2.3}$$

Keterangan:
Env(l) = sinyal envelope.
N_{hop} = sampel antara *successive non-overlapping*.
N_w = panjang dari *frame* dalam beberapa waktu.
n = indeks waktu.
s = spektrum yang diekstak dari *frame* ke-*l.*
L = jumlah total dari *frame.*

2.5.4. *Log Attack Time*

Log Attack Time (LAT) adalah waktu yang dibutuhkan untuk mencapai amplitudo maksimal dari waktu batas minimum. Tujuannya adalah sebagai deskripsi onset sampel suara tunggal dari alat musik yang berbeda. Nilainya didefinisikan sebagai log(basis desimal) dari durasi waktu T_{Start} ketika sinyal mulai dan T_{Stop} ketika mencapi nilai maksimum. Persamaan 2.4 adalah cara mendapatkan nilai LAT.

$$LAT = \log_{10}(T_{stop} - T_{start}) \tag{2.4}$$

2.5.5. *Audio Spectrum Centroid*

Audio Spectrum Centroid (ASC) merepresentasikan sebagai karakteristik dari sebuah spektrum. Bisa juga menunjukkan pusat dari sebuah spektrum. Secara perseptual, ASC memiliki

hubungan kuat antara kejernihan suara. Persamaan 2.5 adalah cara mendapatkan nilai ASC.

$$ASC = \frac{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} \log_2\left(\frac{f'(k')}{1000}\right) P'(k')}{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} P'(k')} \tag{2.5}$$

di mana tiap koefisien yang dibutuhkan didapatkan pada Persamaan 2.6, Persamaan 2.7, Persamaan 2.8, dan Persamaan 2.9.

$$K_{low} = floor(\frac{6.25}{\Delta F}) \tag{2.6}$$

$$P'(k') = \left\{ \sum_{k=0}^{K_{low}} P(k), \qquad for\ k' = 0 \right\} \tag{2.7}$$

$$P'(k') = \begin{cases} \sum_{k=0}^{K_{low}} P(k), & for\ k' = 0 \\ P(k' + K_{low}), & for\ 1 \le k' \le N_{FT} - K_{low} \end{cases} \tag{2.8}$$

$$f'(k') = \begin{cases} 32.25 & for\ k' = 0 \\ f(k' + K_{low}) & for\ 1 \le k' \le \frac{N_{FT}}{2} - K_{low} \end{cases} \tag{2.9}$$

keterangan:
N_{FT} = ukuran dari *Fast Fourier Transform.*
P(k) = power spektrum yang diekstrak pada *frame* ke-*l*.
ΔF = interval frekuensi dari dua sinyal FFT.
k = frekuensi indeks.
f'(k') = frekuensi yang sesuai dengan indeks *k'*.

2.5.6. *Audio Harmonicity*

Audio Harmonicity adalah fitur yang mendeskripsikan 2 properti sinyal harmonik dari spektrum.Properti pertama *harmonic ratio*, yaitu rasio daya harmonik dari total daya dan yang kedua *upper limit harmonicity* yaitu frekuensi spektrum yang tidak dapat dianggap harmonis. Tujuannya untuk membedakan suara harmonik (alat musik contohnya) dan suara non-harmonik (*noise*, pidato tidak jelas, dsb).

2.5.7. *Harmonic Spectral Centroid*

Harmonic Spectral Centroid (HSC) adalah rata-rata dari durasi sinyal amplitudo *weighted mean* dari puncak harmonik spektrum. Nilainya diperoleh dari rata-rata jumlah total *frame centroid*. HSC bisa juga didefinisikan sebagai ukuran ketajaman sinyal timbral. Persamaan 2.10 adalah cara mendapatkan nilai HSC.

$$HSC = \frac{1}{L}\sum_{l=0}^{L-1} LHSC, \tag{2.10}$$

di mana nilai LHSC diperoleh dari Persamaan 2.11.

$$LHSC_l = \frac{\sum_{h=1}^{N_h}(f_{h,l}A_{h,l})}{\sum_{h=1}^{N_h} A_{h,l}} \tag{2.11}$$

Keterangan:

$f_{h,l}$ = frekuensi.
$A_{h,l}$ = amplitudo puncak harmonik.
N_h = jumlah harmonik yang diperhitungkan
L = jumlah frame dalam segmen suara.
$LHSC_l$ = lokal *Harmonic Spectral Centroid*.
h = frekuensi dasar.

2.5.8. *Harmonic Spectral Deviation*

Harmonic Spectral Deviation (HSD) adalah fitur yang mengukur deviasi dari puncak harmonik dari envelope dari spektrum lokal. Persamaan 2.12 adalah cara mendapatkan nilai HSD.

$$HSD = \frac{1}{L}\sum_{l=0}^{L-1} LHSD_l \tag{2.12}$$

di mana nilai $LHSD_l$ diperoleh dari Persamaan 2.13.

$$LHSD_l = \frac{\sum_{h=1}^{N_h}\left|\log_{10}(A_{h,l}) - \log_{10}(SE_{h,l})\right|}{\sum_{h=1}^{N_h}\log_{10}(A_{h,l})} \tag{2.13}$$

$SE_{h,l}$ adalah perkiraan interpolasi amplitudo puncak harmonik yang berdekatan $A_{h,l}$ seperti yang ditulis pada Persamaan 2.14.

$$SE_{h,l} = \begin{cases} \frac{1}{2}\left(A_{h,l} + A_{h+1,l}\right) \; if \; h = 1 \\ \frac{1}{3}\left(A_{h-1,l} + A_{h,l} + A_{h+1,l}\right) \; if \; 2 \le h \le N_H - 1. \\ \frac{1}{2}\left(A_{h-1,l} + A_{h,l}\right) \; if \; h = N_H \end{cases} \tag{2.14}$$

Keterangan:
$A_{h,l}$ = amplitudo puncak harmonik.
N_H = jumlah harmonik yang diperhitungkan.
L = jumlah frame dalam segmen suara.
$LHSC_l$ = lokal *Harmonic Spectral Centroid*.
h = indeks dari komponen harmonis.

2.5.9. *Harmonic Spectral Spread*

Harmonic Spectral Spread (HSS) merepresentasikan ukuran spektrum rata-rata menyebar dalam kaitan dengan HSC. Didefinisikan juga sebagai penyimpangan kekuatan *weighted-RMS* dari HSCL HSCl lokal. Persamaan 2.15 adalah cara mendapatkan nilai HSS.

$$HSS = \frac{1}{L}\sum_{l=0}^{L-1} LHSS_l \tag{2.15}$$

di mana nilai LHSS diperoleh dari Persamaan 2.16.

$$LHSS_l = \frac{1}{LHSS_l}\sqrt{\frac{\sum_{h=1}^{N_H}\left[\left(f_{h,l} - LHSC_l\right)^2 A_{h,l}^2\right]}{\sum_{h=1}^{N_H} A_{h,l}^2}} \tag{2.16}$$

Keterangan:
L = jumlah frame dari segment suara.
$LHSS_l$ = cerminan modulasi getaran yang kurang jelas dari $LHSD_l$.
$f_{h,l}$ = frekuensi.
$A_{h,l}$ = amplitudo puncak harmonik.
N_H = jumlah harmonik yang diperhitungkan
L= jumlah frame dalam segmen suara.
$LHSC_l$ = dijelaskan pada Persamaan 2.11.

2.5.10. *Harmonic Spectral Variation*

Harmonic Spectral Variation (HSV) mencerminkan variasi spektral antara frame yang berdekatan. HSV didefinisikan juga sebagai pelengkap untuk korelasi normalisasi antara amplitudo puncak harmonik yang diambil dari dua frame yang berdekatan. Persamaan 2.17 adalah cara mendapatkan nilai HSV.

$$HSV = \frac{1}{L}\sum_{l=0}^{L-1} LHSV_l \tag{2.17}$$

di mana nilai $LHSV_l$ diperoleh dari Persamaan 2.18.

$$LHSV_l = 1 - \frac{\sum_{h=1}^{N_H}(A_{h,l-1}\, A_{h,l})}{\sqrt{\sum_{h=1}^{N_H} A_{h,l-1}^2}\ \sqrt{\sum_{h=1}^{N_H} A_{h,l}^2}} \tag{2.18}$$

Keterangan:
L = jumlah frame dari segment suara.
$A_{h,l}$ = amplitudo puncak harmonik.
N_H = jumlah harmonik yang diperhitungkan.
h = indeks dari komponen harmonis.

2.5.11. *Audio Spectrum Spread Type*

Audio Spectrum Spread (ASS) didefinisikan sebagai momen sentral kedua spektrum log-frekuensi. Fitur ini diekstraksi dengan mengambil *Root Mean Square* (RMS) dari penyimpangan spektrum dari *Audio Spectrum Centroid*. Persamaan 2.19 adalah cara mendapatkan nilai ASS.

$$ASS = \sqrt{\frac{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} \left[\log_2\left(\frac{f'(k')}{1000}\right) - ASC\right]^2 P'(k')}{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} P'(k')}} \tag{2.19}$$

di mana:
N_{FT} = ukuran dari *Fast Fourrier Transform.*
P'(k') = *power* spektrum yang diekstrak pada *frame* ke-*l.*
k = frekuensi indeks.

ASC = dijelaskan pada Persamaan 2.5.
K_{low} = dijelaskan pada Persamaan 2.6.
f '(k') = dijelaskan pada Persamaan 2.9.

2.5.12. *Audio Spectrum Projection*

Audio Spectrum Projection (ASP) merupakan fitur yang didapat dari teknik normalisasi *Singular Value Decomposition* (SVD) dan Independent Component Analysis (ICA). Teknik ini diterapkan untuk proses klasifikasi, identifikasi alat musik, dan lain-lain.

2.5.13. *Audio Spectrum Basis Type*

Audio Spectrum Basis (ASB) didefinisikan sebagai representasi proyeksi spektrum dimensi rendah, sistem klasifikasi. Ekstrasi fitur ini menggunakan teknik normalisasi standar *Singular Value Decomposition* (SVD) dan *Independent Component Analysis* (ICA).

2.5.14. *Audio Spectrum Flatness Type*

Audio Spectrum Flatness (ASF) didefinisikan sebagai cerminan kerataan properti suatu kekuatan spektrum. Persamaan 2.20 adalah cara mendapatkan nilai ASF.

$$ASF = \frac{\sqrt[hiK'_b - loK'_b + 1]{\prod_{k'=loK'_b}^{hiK'_b} P_g(k')}}{\frac{1}{hiK'_b - loK'_b + 1} \sum_{k'=loK'_b}^{hiK'_b} P_g(k')} \qquad (1 \le b \le B). \tag{2.20}$$

di mana tiap koefisien yang dibutuhkan didapatkan pada Persamaan 2.21, Persamaan 2.22, Persamaan 2.23, dan Persamaan 2.24.

$$loEdge = 2^{\frac{1}{4}N} \times 1\ kHz \tag{2.21}$$

$$hiEdge = 2^{\frac{1}{4}B} \times loEdge \tag{2.22}$$

$$loF_b = 0.95 \times loEdge \times 2^{\frac{1}{4}(b-1)} \tag{2.23}$$

$$hiF_b = 1.05 \times loEdge \times 2^{\frac{1}{4}b} \tag{2.24}$$

Keterangan:
B = nilai penentu batas atas *band.*
b = frekuensi dari indeks *band.*
$P_g(k')$ = *power* spektrum yang diekstrak pada *frame* ke-*l*.
hiK_b = frekuensi *integer* batas dari hiF_b.
loK_b = frekuensi *integer* batas dari loF_b.
band = 1 kHz.

2.5.15. *Audio Spectrum Envelope Type*

Audio Spectrum Envelope (ASE) adalah Fitur yang mendeskripsikan jumlah dari koefisien *power band* b. ASC diperoleh dengan menjumlahkan *power* spektrum dalam serangkaian frekuensi *bands*. Persamaan 2.25 adalah cara mendapatkan nilai ASE.

$$ASE(b) = \sum_{k=loK_b}^{hiK_b} P(k) \quad (1 \leq b \leq B_{in}), \tag{2.25}$$

di mana tiap koefisien yang dibutuhkan didapatkan pada Persamaan 2.26, Persamaan 2.27, dan Persamaan 2.28.

$$r = 2^j octaves \ (-4 \leq j \leq +3) \tag{2.26}$$

$$loF_b = loEdge \times 2^{(b-1)r} \ (1 \leq b \leq B_{in}) \tag{2.27}$$

$$hiF_b = loEdge\ 2^{br} (1 \leq b \leq B_{in}) \tag{2.28}$$

Keterangan:
loEdge = 62.5 Hz.
$B_{in} = \frac{8}{r}$.
b = frekuensi dari indeks *band.*
k = frekuensi indeks.
hiK_b = frekuensi *integer* batas dari $hiF_.$
loK_b = frekuensi *integer* batas dari $loF_{b.}$

2.5.16. *Audio Fundamental Frequency*

Audio Fundamental Frequency (AFF) merupakan fitur yang memberikan perkiraan mendasar frekuensi segmen f_0 di mana sinyal diasumsikan periodik.

Ada parameter yang ditambahkan dalam standarisasi MPEG-7 ini, yaitu:

- *loLimits*: batas bawah dari rentang frekuensi di mana f_0 telah dicari.
- *hiLimits*: batas atas dari rentang frekuensi di mana f_0 telah dicari.

Ukuran kepercayaan pada kehadiran periodisitas di bagian sinyal nilainya antara 0 dan 1.

2.5.17. *Audio Signature*

Audio Signature (AS) adalah suatu fitur yang menjadi mencerminkan identik suatu lagu. AS bisa juga disebut sebagai *fingerprint* dari sebuah lagu. Tujuannya untuk *matching* suatu sinyal identik dengan lagu apa.

2.5.18. *Sound Model*

Sound Model adalah Fitur yang digunakan untuk menghitung suatu *state sound model* histrogram.

2.6. *Fast Fourier Transform*

Fast Fourier Transform (FFT) adalah metode untuk mengubah sinyal dari *time domain* menjadi *frequency domain* [10]. Tujuannya adalah untuk mencari suatu informasi penting dalam *frequency domain* untuk tahap analisis [11][12].

2.7. *Discrete Wavelet Transform*

Discrete Wavelet Transform (DWT) adalah metode wavelet yang digunakan untuk melakukan dekomposisi pada *wavelet* sampai level N [10]. Tujuannya adalah untuk mengurangi *noise* pada sinyal dan memperkuat informasi di dalam sinyal tersebut dengan mempertahankan keutuhan informasi data [13] [14]. Ada banyak metode wavelet, namun dalam pengerjaan Tugas Akhir ini, metode wavelet yang digunakan adalah bior 2.8 dengan mengambil nilai *approximation coefficients*.

Untuk menentukan level dekomposisi ini tidak boleh sembarangan karena ketika level dekomposisi tinggi belum tentu yang baik, sebaliknya malah merusak sinyal sehingga menghilangkan informasi yang terkandung sinyal asli. Maka perlu dilakukan pemilihan level dekomposisi wavelet yang terbaik [15].

Langkah pertama adalah melakukan perhitungan pada Persamaan 2.29 yang dilakukan pada Matlab.

$$[\max value, index\ max] = max\left(abs\left(FFT(S - mean(S))\right)\right) \tag{2.29}$$

Maka akan didapatkan hasil *max value* dan *max index*. Kedua hasil ini digunakan untuk mencari *Fh*. *Fh* adalah *Frequency Range* pada Tabel 2.1. Untuk mendapatkan *Fh*, lakukan perhitungan pada Persamaan 2.30.

$$Fh = index\max * Fs/L \tag{2.30}$$

Di mana *Fs* adalah frekuensi sampling yaitu 1024, dan L adalah panjang sinyal. Hasil dari *Fh* dapat dilihat dari Tabel 2.1 sesuai dengan *Frequency Range.* Aturan untuk menentukan tingkat dekomposisi pada Tabel 2.1 dapat dinyatakan oleh Persamaan 2.31 [10].

$$\frac{f_q}{2^N + 1} \leq f_{char} \leq \frac{f_q}{2^N} \tag{2.31}$$

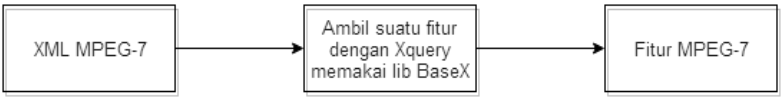
di mana f_q adalah *sampling frequency*, f_{char} adalah *dominant frequency*, dan *N* adalah level dekomposisi. Maka didapatlah level *decompose wavelet* terbaik.

Tabel 2.1 Penentuan Decompose Level Wavelet

***Decomposition Level* (L)**	***Frequency Range* (Hz)**
1	256-512
2	128-256
3	64-128
4	32-64
5	16-32
6	8-16
7	4-8
8	2-4
9	1-2
10	0.5-1
11	0.25-0.5
12	0.125-0.25
13	0.0625-0.125

2.8. *Xquery*

Xquery adalah bahasa untuk meng-*query* atau pemanggilan data di dalam suatu *database* dalam bentuk file XML [16]. Pada Tugas Akhir ini, *Xquery* digunakan untuk mengambil suatu fitur pada XML yang dihasilkan lewat ekstraksi fitur MPEG-7. Untuk mengimplementasikan *Xquery* menggunakan *library* Java yang bernama *BaseX* [17]. Gambar 2.8 adalah tahapan proses yang dilakukan menggunakan *Xquery*. *Input*nya adalah suatu XML yang dihasilkan dari ekstraksi audio fitur berbasis MPEG-7 dan *output*nya berupa file teks yang berisi fitur yang ingin diambil.



Gambar 2.8 Tahapan Proses Xquery

2.9. *Valence*

Valence adalah suatu nilai yang menggambarkan seberapa tingkat kesenangan/*enjoy* seseorang terhadap sesuatu. Skala nilai rentang dari 1-9. Semakin tinggi angka *valence* maka menunjukan semakin senang emosi seseorang tersebut.

2.10. *Arousal*

Arousal adalah suatu nilai yang menggambarkan seberapa tingkat ketegangan/aktif seseorang terhadap sesuatu. Skala nilai rentang dari 1-9. Semakin tinggi angka *arousal* maka menunjukan semakin aktif/tegang emosi seseorang tersebut.

2.11. *Dataset*

Dataset adalah kumpulan sutu data yang dilakukan untuk uji coba. Pada uji coba yang dilakukan untuk Tugas Akhir ini diambil dari *1000 Songs for Emotional Analysis of Music* [18] [19]. Dari *dataset* ini sudah terdapat nilai *valence* dan *arousal* sebagai penentuan *mood*, *genre* sebuah lagu, dan BPM lagu yang pasti dimiliki semua lagu.

2.12. *Mood*

Mood adalah keadaan emosional yang bersifat sementara, bisa beberapa menit bahkan beberapa minggu. *Mood* juga bisa diartikan tanggapan kita terhadap suatu rangsangan yang terjadi. *Mood* berbeda dengan emosi. Emosi adalah perasaan intens yang diarahkan pada seseorang atau sesuatu. Sedangkan *mood* adalah perasaaan yang tumbuh kurang intens yang dikarenakan kekurangan suatu stimulus. Secara umum, perbedaan emosi dan *mood* dipetakan pada Tabel 2.2 [20].

Tabel 2.2 Perbedaan Antara Emosi dan Mood

Emosi	Mood
Durasi singkat	Durasi lama
Spesifik mengarah kepada suatu hal	Biasanya lebih umum
Biasanya disertai dengan beragam ekspresi wajah	Biasanya tidak disertai dengan ekspresi wajah

Pada musik, yang dipakai pada uji coba kali ini adalah yang *mood*, dikarenakan musik adalah stimulus yang bagus untuk meningkatkan *mood* seseorang.

2.13. *Genre*

Genre musik adalah pengelompokan musik sesuai dengan kemiripan satu sama lain. Sebuah *genre* dapat juga didefinisikan oleh teknik musik, gaya, dan konteks musik.

Terdapat banyak macam variasi *Genre* pada suatu musik [21]. Dikarenakan sebuah *genre* pada suatu lagu terlalu banyak, maka pada percobaan ini dibatasi hanya sebatas *genre* umum suatu lagu, yaitu *classic, electronic, jazz,* dan *rock* dimana di dataset juga sudah tersedia. *Genre classic* adalah musik yang memiliki ciri peralihan nada dari awalan lembut menjadi keras atau sebalikya. *Genre electronic* adalah musik yang menggunakan alat musik elektronik dan teknologi musik untuk produksinya. *Genre jazz* adalah musik yang sebagian besar lagunya didominasi oleh gitar,

trombon, piano, terompet, dan saksofon. *Genre rock* adalah jenis musik yang berpusat pada permainan gitar listrik, bass listrik, dan drum.

2.14. Tempo

Tempo musik adalah ukuran kecepatan dalam birama lagu. Ukuran kecepatan bisa diukur dengan alat bernama *metronome* dan *keyboard*. Secara umum, tempo dibagi menjadi 3 yaitu lambat, sedang, dan cepat. Tempo dibagi berdasarkan kecepatan nada [22]. Tabel 2.3 adalah pemetaan pembagian tempo berdasarkan *Beats Per Minutes* (BPM).

Tabel 2.3 Pembagian Estimasi Tempo

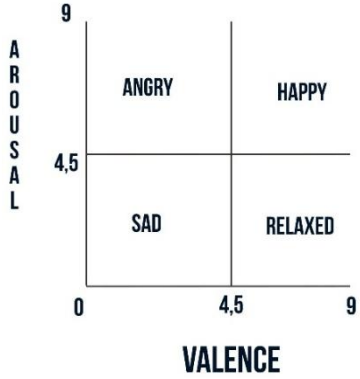
Satuan BPM	Estimasi Tempo
<100	Lambat
100-135	Sedang
>135	Cepat

Untuk mendapatkan BPM suatu lagu, penelitian ini menggunakan *software* yang bernama MixMeister BPM Analyzer yang dapat diunduh secara gratis [23]. Langkah-langkah mendapatkan suatu lagu dapat dilihat pada Gambar 2.9. Pertama buka program MixMeister BPM Analyzer, lalu *drag* suatu lagu yang ingin dicari nilai BPM-nya. Maka program akan otomatis menghitung berapa nilai yang dimiliki suatu lagu. Nilai BPM inilah yang akan digunakan untuk proses klasifikasi untuk menentukan *slow*, *medium*, maupun *fast*.

Gambar 2.9 Tahapan Mendapat BPM Suatu Lagu

2.15. *Russel's Circumplex models*

Russel's Circumplex models adalah model yang digunakan untuk memetakan emosi dari dua kombinasi linear:*arousal* dan *valence* [24]. Pada Tugas Akhir ini, akan dipetakan berdasarkan skor *arousal* dan *valence* sesuai dengan Gambar 2.10.



Gambar 2.10 Bagan Russel's Circumplex

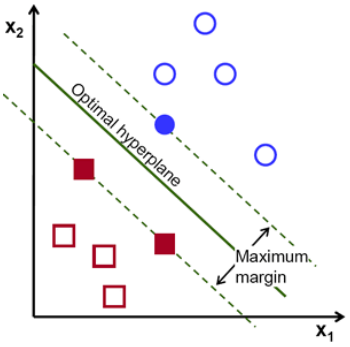
2.16. *Support Vector Machine*

Support Vector Machine (SVM) adalah metode *machine learning* yang bekerja dengan mencari *hyperlane* terbaik *(*fungsi klasifier) yang memisahkan beberapa label yang berbeda [1]. *Hyperlane* yang optimal dapat ditemukan dengan cara mengukur *margin/distance* antara *hyperlane* dengan data yang paling dekat pada masing-masing labelnya, seperti pada contoh Gambar 2.11.

SVM telah digunakan dalam berbagai eksperimen seperti klasifikasi identifikasi pembicara, pengenalan obyek, deteksi wajah dan klasifikasi vokal. SVM dapat mengklasifikasikan data multi-dimensi adalah yang pada dasarnya menentukan perbatasan antara dua kelas atau lebih.

Contoh pelatihan *training* data pada SVM menentukan parameter dari keputusan untuk mengklasifikasikan fungsi dari dua atau lebih kelas dan memaksimalkan margin selama fase

pembelajaran. Setelah tahap pembelajaran, mesin dapat memperkirakan pola yang tidak diketahui untuk diklasifikasikan.



Gambar 2.11 Proses Klasifikasi SVM

Pada penelitian ini, dilakukan sesi *training* dan sesi *testing* untuk mencoba klasifikasi. Sesi *training* merupakan sesi melakukan pembelajaran terhadap variasi data-data yang dimiliki oleh suatu label sedangkan sesi *testing* merupakan proses uji coba prediksi pada suatu data yang baru. Metode SVM dipilih karena percobaan sebelumnya klasifkasi SVM menggunakan fitur MPEG-7 dan hasilnya baik [25]. Untuk melakukan implementasi SVM ini, digunakan *library sklearn* dari bahasa pemrograman Python menggunakan *default kernel* yaitu rbf.

[Halaman ini sengaja dikosongkan]

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis dan perancangan sistem yang akan dibangun. Analisis membahas semua persiapan yang akan menjadi pokok pikiran pembuatan aplikasi ini. Mulai dari masalah yang melatarbelakangi, hingga analisis gambaran awal sistem yang akan dibuat. Perancangan sistem membahas hal-hal yang berkaitan dengan pondasi atau dasar pembuatan aplikasi, yang meliputi perancangan basis data, tampilan antar muka halaman aplikasi, hingga perancangan alur proses yang akan diimplementasikan di dalam aplikasi.

3.1. Analisis

Tahap analisis meliputi analisis masalah, analisis kebutuhan, deskripsi umum sistem, dan kasus penggunaan sistem yang dibuat.

3.1.1. Kontribusi Penelitian

Sebenarnya sudah ada penelitian-penelitian terkait mengenai MIR menggunakan MPEG-7 ini. Salah satu contohnya adalah deteksi perbedaan suara (*Sound Recognition*) pada kucing, kodok, senapan, dan lain lain [25]. Namun untuk klasifikasi *mood* dan tempo pada suatu lagu masih belum ada sehingga kontribusi dan kebaruan dari penelitian ini adalah *mood classification* dan *tempo estimation* pada suatu audio.

Pada penelitian ini, MPEG-7 digunakan karena ekstraksi fitur yang dihasilkan berupa metadata. Di dalam metadata tersebut berisi angka-angka digital yang merupakan gambaran dari karakteristik-karakteristik sinyal suatu audio sesuai durasinya. Oleh karena itu, penelitian Tugas Akhir ini terkait dengan ilmu *Digital Signal Processing* (DSP).

3.1.2. Analisis Permasalahan

Di era *modern* ini, banyak sekali lagu yang telah diproduksi. Lagu dapat dengan mudah didengarkan seseorang di tempat-tempat umum seperti mal, kafe, toko buku, dsb. Setiap orang pasti mempunyai dafar lagu yang disukainya, namun tidak menutup kemungkinan juga jika seseorang yang sedang berjalan-jalan di suatu tempat umum seperti kafe lalu secara tidak sengaja mendengar lagu baru yang tidak pernah diketahuinya dan langsung menyukainya. Kejadian seperti ini sering dialami banyak orang. Biasanya, seseorang akan mencari lagu tersebut dengan cara mengetik lirik dan mencarinya dalam sebuah mesin pencarian. Namun bagaimana ketika sebuah lagu tersebut susah untuk dicari dalam mesin pencarian? Misalnya, bahasa yang susah (karena bahasa asing), lirik yang dinyanyikan terlalu cepat, dsb. Hal ini membuat seseorang gagal mendapatkan lagu yang baru diinginkan sehingga terkadang timbul perasaan kecewa.

Oleh karena itu, aplikasi MusicMoo ini dibuat untuk menangani permasalahan di atas. Ketika seseorang tertarik dengan lagu baru yang didengarnya, cukup membuka aplikasi perangkat bergerak MusicMoo dan merekam lagu setidaknya 45 detik. Maka aplikasi akan memberikan judul lagu yang dimaksud ataupun detail informasi lagu seperti *mood*, *genre*, dan tempo.

3.1.3. Analisis Kebutuhan

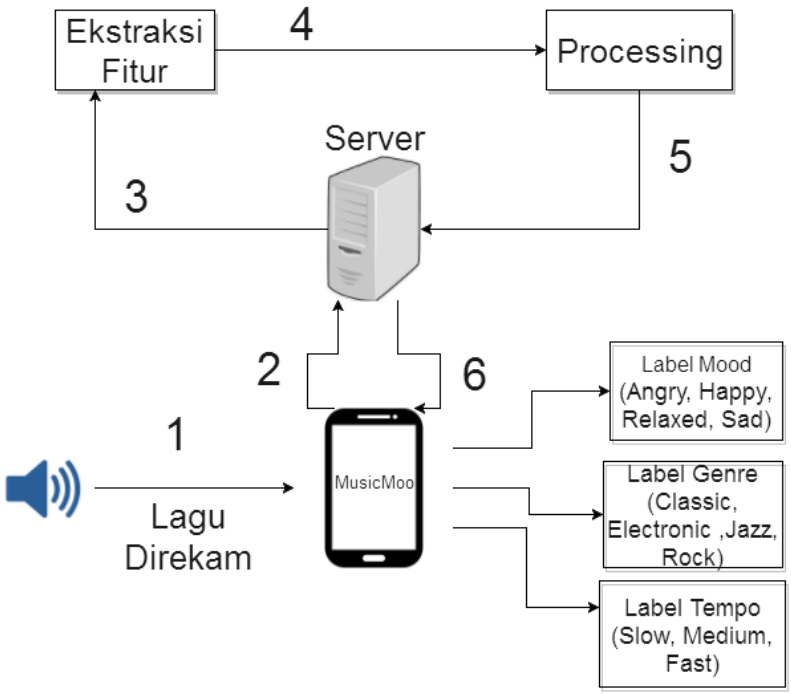
Kebutuhan utama dalam aplikasi ini difokuskan untuk mendeteksi *genre*, *mood*, serta tempo yang dimiliki oleh suatu lagu. Contoh aplikasi yang sudah ada adalah SoundHound dan Shazam. Kedua aplikasi ini mampu mendeteksi potongan lagu tersebut berjudul apa, akan tetapi masih terbatas hanya judul lagu tanpa informasi detail seperti *mood*, *genre*, maupun tempo. Oleh karena itu, dengan aplikasi yang dirancang bertujuan untuk menangani batasan yang dimiiki oleh SoundHound dan Shazam tersebut. Tabel 3.1 merupakan daftar kebutuhan fungsional perangkat lunak yang dibangun sesuai dengan modul.

Tabel 3.1 Kebutuhan Fungsional Sistem

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
F-1	Mendeteksi *mood* lagu	Partisipan dapat mengenali sebuah lagu memiliki *mood* apa.
F-2	Mendeteksi *genre* lagu	Partisipan dapat mengenali sebuah lagu memiliki *genre* lagu apa.
F-3	Mendeteksi tempo lagu	Partisipan dapat mengenali sebuah lagu tersebut bertempo apa.

3.1.4. Deskripsi Umum Sistem

Aplikasi yang akan dibuat pada Tugas Akhir ini adalah program aplikasi perangkat bergerak. Gambar 3.1 adalah gambaran alur jalan sistem.



Gambar 3.1 Deskripsi Umum Sistem

Keterangan pada penomoran Gambar 3.1:

1. Sebuah lagu direkam melalui aplikasi perangkat bergerak.
2. Lagu rekaman di-*upload* ke server.
3. Lagu rekaman dilakukan ekstraksi fitur dan pengambilan fitur oleh server Java.
4. Fitur yang diambil dilakukan *processing* pada server Python (melakukan DWT dan klasifikasi).
5. Hasil akan dikirim kembali pada server.
6. Server akan menampilkan hasil pada aplikasi perangkat bergerak.

Untuk penjelasan secara rinci akan dijelaskan di bawah mulai dari proses *input*, proses, hingga *output.*

3.1.4.1. *Input*

Pertama-tama, sebuah lagu direkam lewat aplikasi perangkat bergerak. Lakukan perekaman dengan menekan tombol rekam dengan durasi minimal 45 detik. Jika sudah, maka tekan tombol berhenti untuk mengakhiri rekaman. Maka hasil rekaman akan tersimpan dengan format .wav. Berikutnya adalah tekan tombol kirim. Maka lagu akan terunggah ke server dan siap diproses.

3.1.4.2. Proses

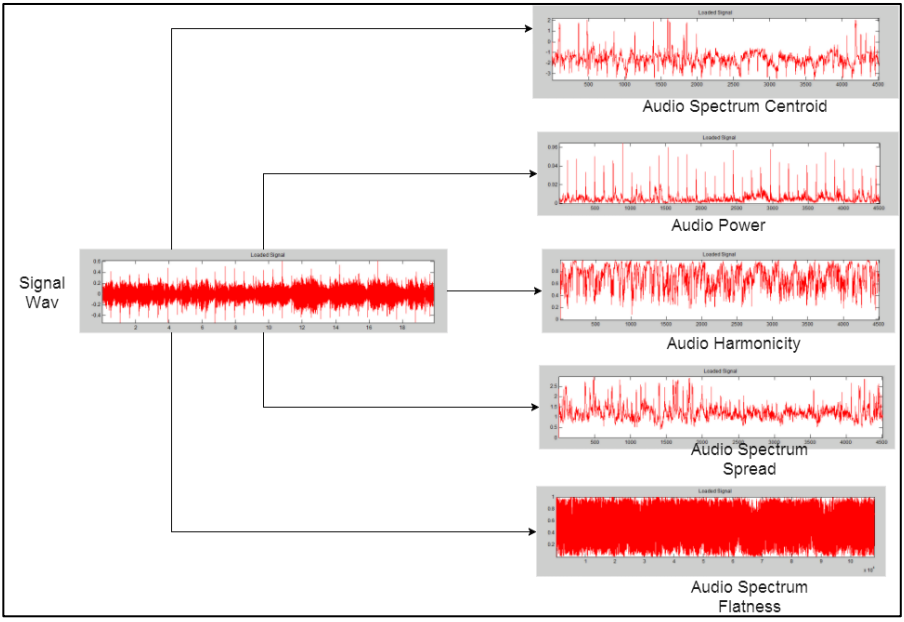
Pada tahap ini dijelaskan secara bertahap langkah-langkah yang dilakukan pada sistem dimulai dari proses ekstraksi fitur, pengolahan sinyal fitur, sampai menjadi hasil *output*.

3.1.4.2.1 Ekstraksi Fitur

Setelah lagu ter*upload*, maka akan dilakukan ekstraksi fitur berbasis MPEG-7 yang dilakukan pada server Java. Pada tahap ini, tujuannya adalah untuk mendapatkan XML MPEG-7 dari sebuah rekaman lagu dan mengambil fitur yang digunakan untuk proses klasifikasi. Ektraksi fitur ini memakai *library* pada Java yang bernama *MPEG7AudioEnc* dan untuk mengambil fitur

dari XML yang dihasilkan menggunakan *Xquery* yang diimplementasikan oleh *library* Java yang bernama *BaseX*.

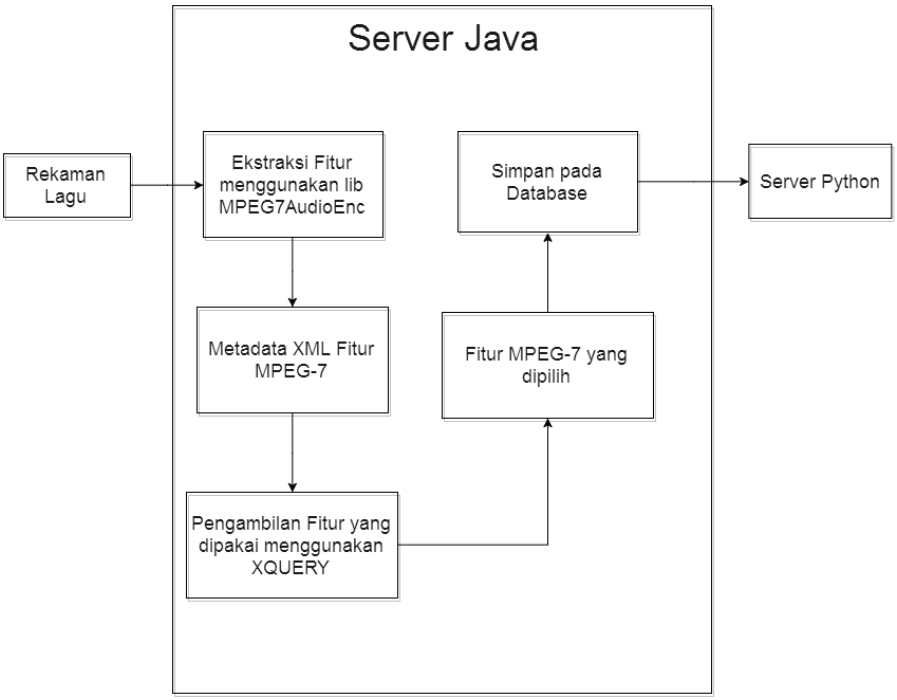
Gambar 3.2 adalah ilustrasi bagaimana dari wav utuh dilakukan ekstraksi fitur. Suatu sinyal wav dipecah-pecah menjadi karakteristik yang menggambarkan variasi-variasi dari sinyal penuh. Fitur-fitur inilah yang akan digunakan untuk proses klasifikasi. Pada proses klasifikasi, tidak semua fitur akan dipakai, hanya yang berpengaruh sajalah yang diambil dan dipakai untuk tahap *processing.*



Gambar 3.2 Ekstaksi Sinyal Wav

Untuk detail tahapan proses yang terjadi di server Java digambarkan pada Gambar 3.3 di mana suatu lagu akan dilakukan ekstraksi fitur sehingga menghasilkan metadata XML berisi fitur-fitur MPEG-7. Setelah metadata dihasilkan, maka dilakukan

pengambilan fitur oleh *Xquery* untuk mengambil fitur-fitur yang akan dipakai untuk disimpan dalam *database*. Jika sudah maka tugas pada server Java selesai dan dilanjutkan tahap *processing* yang ditangani oleh server Python.



Gambar 3.3 Tahapan Ekstraksi Fitur Pada Server Java

Untuk pemilihan fitur yang dipakai dan disimpan berdasarkan teori sebagai berikut:

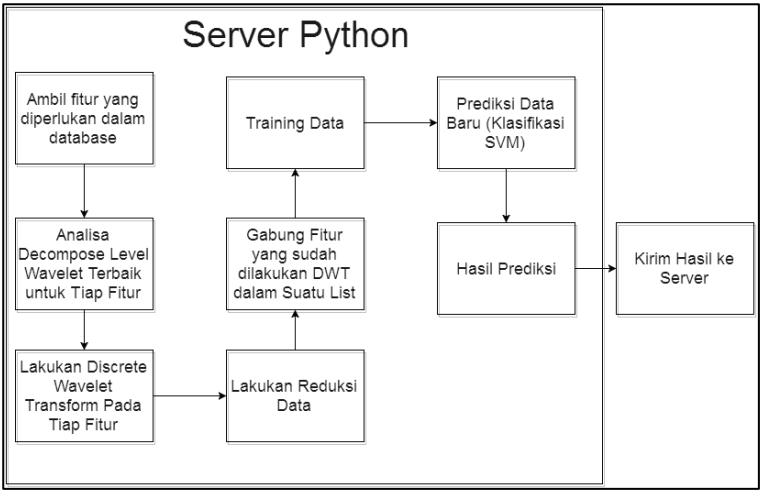
1. Pada modul *mood*, audio fitur yang dipakai untuk proses adalah *Audio Power* dan *Audio Harmonicity*. Kedua fitur ini dipilih dan dikombinasikan karena untuk *mood* suatu lagu, dipengaruhi oleh suatu daya/naik turunnya amplitudo

serta keselarasan harmonis nada [26]. Maka dipilihlah fitur *Audio Power dan Harmonicity*.

2. Pada modul *genre* dan tempo, audio fitur yang dipakai untuk proses adalah *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness* [27]. Fitur ini dipakai karena suatu lagu yang mempengaruhi *genre* dan tempo adalah informasi kejernihan suara (ASC), penyimpangan spektrum dari sinyal asli (ASS), dan kerataan properti suatu kekuatan spektrum (ASF). Maka untuk *data train* adalah gabungan ketiga sinyal yaitu ASC, ASS, dan ASF.

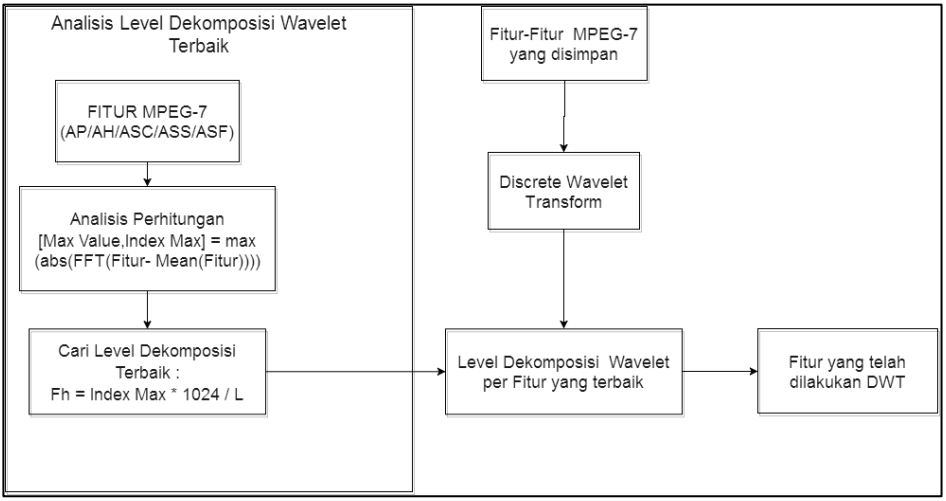
3.1.4.2.2 *Processing*

Pada tahap ini, fitur yang dipilih akan masuk ke tahap *processing*. Fitur yang dipilih ini akan dilakukan *Discrete Wavelet Transform* (DWT) dengan menggunakan tipe *wavelet bior* 2.8. Tujuannya adalah untuk menghilangkan *noise* yang terdapat pada sinyal. Implementasi yang dilakukan untuk melakukan DWT ini menggunakan *library* Python yang bernama *Pywt*. Untuk detail tahapan proses yang terjadi di server Python digambarkan pada Gambar 3.4.



Gambar 3.4 Tahapan Processing Pada Server Python

Langkah pertama yang dilakukan server Python adalah mengambil seluruh fitur-fitur yang sudah disimpan dalam *database*. Kemudian fitur-fitur inilah yang akan dilakukan proses DWT. Pada tahap melakukan DWT ini terdapat langkah-langkah yang dipaparkan pada Gambar 3.5.

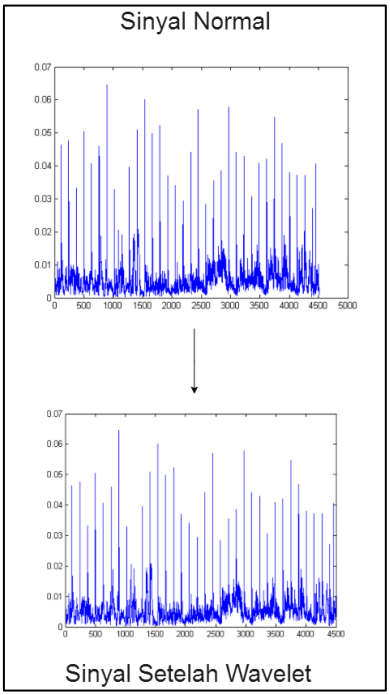


Gambar 3.5 Tahapan Melakukan DWT

Tahapan untuk melakukan DWT sebagai berikut:

1. Menentukan level dekomposisi fitur-fitur yang akan digunakan (*Audio Power*, *Audio Harmonicity*, *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*).
2. Lakukan perhitungan pada step analisis perhitungan.
3. Cari level dekomposisi sesuai dengan rentang frekuensi yang terdapat pada Tabel 2.1.
4. Maka didapatkan level dekomposisi wavelet terbaik untuk tiap fitur.
5. Lakukan proses DWT untuk semua fitur yang akan dipakai sesuai dengan level dekomposisinya.

Contoh sinyal yang telah dilakukan wavelet dengan level dekomposisi terbaik pada Gambar 3.6. Gambar bagian atas adalah sinyal normal sedangkan bagian bawah adalah sinyal yang sudah terwavelet. Dari gambar tersebut menunjukan bahwa sinyal yang telah dilakukan DWT tidak merusak sinyal aslinya.



Gambar 3.6 Contoh Sinyal Setelah Terwavelet

Kemudian dilanjutkan tahapan pada server Python yaitu reduksi data. Reduksi data yang dimaksud adalah menyamakan panjang sinyal agar menjadi seragam untuk proses klasifikasi. Hal ini dilakukan karena ekstraksi fitur yang dihasilkan mempertimbangkan milidetik juga. Pada penelitian ini, telah dilakukan analisis pada sejumlah 296 lagu dari dataset dan

menganalisis panjang data semua fitur yang sudah dilakukan DWT. Tujuannya adalah untuk mengambil panjang lagu minimal untuk dijadikan acuan penyamarataan panjang sinyal. Tabel 3.2 adalah hasil analisis panjang sinyal minimal untuk tiap fitur.

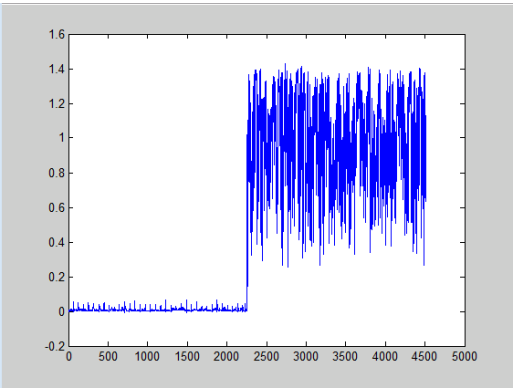
Tabel 3.2 Panjang Minimal Tiap Fitur

Fitur	Panjang minimal
Audio Power	4.498
Audio Harmonicity	4.493
Audio Spectrum Centroid	51
Audio Spectrum Spread	51
Audio Spectrum Flatness	107.904

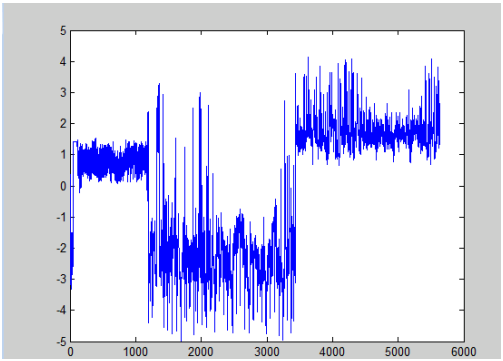
Maka panjang sinyal fitur setiap musik yang diambil sebanyak angka-angka tersebut, sisanya bisa diabaikan. Proses ini dijamin tidak akan merusak sinyal karena dari data awal panjang sinyal seragam semua yaitu 45 detik dan hanya membuang milidetik saja.

Setelah proses ini, akan masuk pada tahap berikutnya yaitu menggabungkan fitur dalam suatu list. Pada tahap ini, fitur-fitur yang sudah dilakukan DWT akan digabungkan menjadi satu *list* untuk menjadi data yamg siap diklasifikasikan.

Untuk modul *mood*, gabungan fitur yang terbentuk berasal dari *Audio Power* dan *Audio Harmonicity*. Sedangkan untuk modul *genre* dan tempo, gabungan fitur yang dibentuk adalah kombinasi dari *Audio Spectrum Centroid, Audio Spectrum Spread,* dan *Audio Spectrum Flatness*. Gambar 3.7 adalah contoh plot *list* sinyal dari kombinasi *Audio Power* dan *Audio Harmonicity* dan Gambar 3.8 adalah contoh plot list sinyal dari kombinasi *Audio Spectrum Centroid, Audio Spectrum Spread,* dan *Audio Spectrum Flatness*. Gabungan-gabungan fitur ini sudah dibagi sama panjang fitur per kolom untuk melakukan klasifikasi, sehingga keaslian informasi sinyal dijamin tidak akan rusak/berbeda.



Gambar 3.7 Gabungan Fitur AP dan AH



Gambar 3.8 Gabungan Fitur ASC, ASS, dan ASF

Terakhir, adalah proses klasifikasi. Metode klasifikasi yang digunakan adalah *Support Vector Machine* (SVM). Sebelum tahap klasifikasi, ada proses yang namanya melatih data. Tujuannya adalah agar mesin dapat mengetahui beragam karakteristik sinyal yang dimaksud sesuai dengan label tertentu. Implementasi SVM dilakukan menggunakan *library* Python yang bernama *sklearn*. Pada uji coba kali ini adalah melakukan *training* data untuk tiap modul (*mood*, *genre*, dan tempo) sebanyak 65 data per label. Tabel 3.3 adalah rincian penjelasan data *training*.

Tabel 3.3 Perincian Data Training

Perincian Data Training		
Modul	**Label**	**Jumlah Data Training**
Mood	Angry	65
	Happy	65
	Relaxed	65
	Sad	65
Genre	Classic	65
	Electronic	65
	Jazz	65
	Rock	65
Tempo	Slow	65
	Medium	65
	Fast	65

Setelah mesin membelajari karakteristik-karakteristik sinyal sesuai yang diberikan, maka prediksi data baru pun dapat dilakukan. Sehingga karakteristik lagu rekaman tadi dapat diprediksi memiliki *mood, genre*, ataupun tempo apa sesuai dengan karakteristik sinyalnya. Hasil prediksi akan dibawa kembali ke server untuk ditampilkan pada aplikasi perangkat bergerak.

3.1.4.3. *Output*

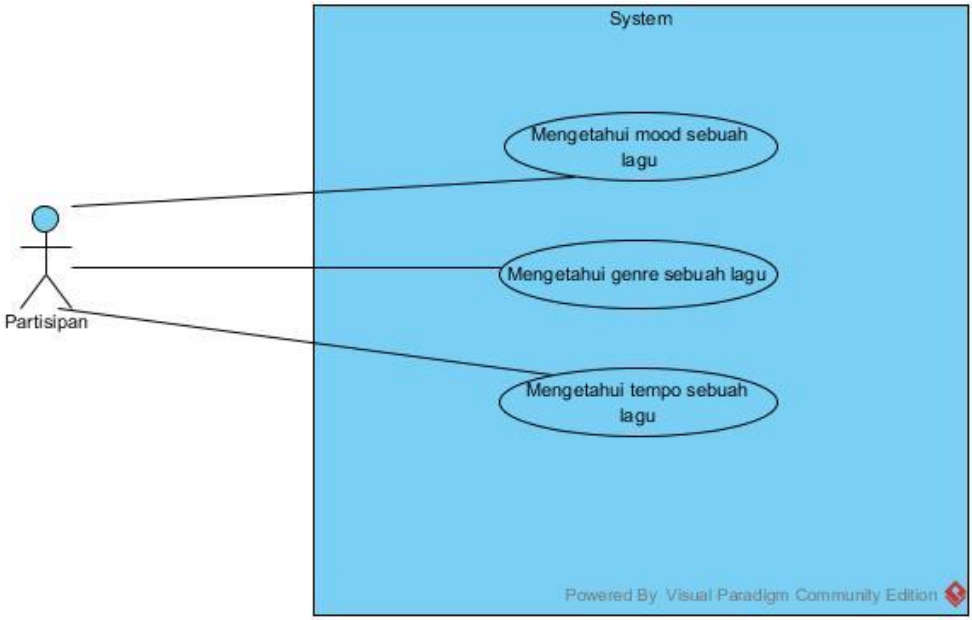
Masuk tahap terakhir yaitu adalah hasil. Lagu yang di-*upload* tadi, akan diprediksi oleh mesin yang sudah 'dilatih'lewat tampilan perangkat bergerak. Untuk *mood, output* suatu lagu dapat berupa label *angry*/*happy*/*relaxed*/*sad.* Pada modul *genre*, hasil berupa label *classic*/*electronic*/*jazz*/*rock*. Terakhir pada modul tempo, hasil berupa label *slow*/*medium*/*fast*. Untuk uji coba, akan diperinci lebih lagi pada Bab V.

3.1.5. Kasus Penggunaan

Mengacu pada spesifikasi kebutuhan fungsional yang telah dipaparkan, dibuat kasus penggunaan yang selanjutnya akan disimpulkan dalam deskripsi umum sistem, yang diharapkan dapat memenuhi kebutuhan fungsional, berdasar pada kasus penggunaan yang dibuat. Kasus penggunaan dijelaskan lebih lanjut pada Tabel 3.4 dan diagram kasus penggunaan ditunjukkan pada Gambar 3.9.

Tabel 3.4 Daftar Kode Kasus Penggunaan

Kode Kasus Penggunaan	Nama	Aktor
UC-1	Mengetahui *mood* sebuah lagu	Partisipan
UC-2	Mengetahui *genre* sebuah lagu	Partisipan
UC-3	Mengetahui tempo sebuah lagu	Partisipan



Gambar 3.9 Diagram Kasus Penggunaan

3.1.4.1. Mengetahui *mood* sebuah lagu (UC-1)

Kasus penggunaan kode UC-1 diakses partisipan setelah berhasil merekam suatu lagu yang dimaksud, lalu diunggah ke server. Pada tahap ini partisipan akan mendapat hasil lagu yang dimaksud berlabel *mood* apa sesuai yang dinilai oleh sistem.

3.1.4.2. Mengetahui *genre* sebuah lagu (UC-2)

Kasus penggunaan kode UC-2 diakses partisipan setelah berhasil merekam suatu lagu yang dimaksud, lalu diunggah ke server. Pada tahap ini partisipan akan mendapat hasil lagu yang dimaksud bertipe *genre* apa sesuai yang dinilai oleh sistem.

3.1.4.3. Mengetahui tempo sebuah lagu (UC-3)

Kasus penggunaan kode UC-3 diakses partisipan setelah behasil merekam suatu lagu yang dimaksud, lalu di diunggah ke server. Pada tahap ini partisipan akan mendapat hasl lagu yang dimaksud memiliki tempo apa sesuai yang dinilai oleh sistem.

3.2. Perancangan Sistem

Tahap ini meliputi perancangan basis data, tampilan antarmuka, dan perancangan alur proses penggunaan sistem yang diharapkan dapat memenuhi tujuan dari pengembangan aplikasi ini. Perlu diketahui bahwa aplikasi ini dibangun dalam kondisi lingkungan tertentu, dan dapat dioperasikan dalam lingkungan tertentu pula. Lingkungan pengembangan aplikasi adalah sebagai berikut.

Perangkat keras	:
– Komputer	: Prosesor Intel® Core™ i3-CPU (3.30GHz), RAM 4 GB, Graphic Intel ® Sandybridge Desktop
– Android	: OS Android v5.1 Lollipop Chipset Mediatek MT6795 Helio X10, Octa-core 2.0 GHz & PowerVR G6200 GPU, RAM: 2 GB

Perangkat lunak :
Sistem operasi : Windows 7, Android Lolipop
IDE : Android Studio 2.0, IntelliJ 2016 1.1
Basis Data : MySQL 5.6
SDK : SDK Android Lolipop level 23.

3.2.1. Perancangan Basis Data

Pada subbab ini dijelaskan mengenai perancangan basis data yang dalam hal ini digunakan untuk menyimpan data diri partisipan beserta rekam skor yang diperolehnya selama menggunakan aplikasi ini. Gambaran perancangan basis data dapat dilihat pada penjelasan di bawah. Saat ini, aplikasi masih menggunakan *database* relasional (MySQL) dikarenakan kekurangan sumber daya manusia, sedangkan seharusnya menggunakan *database* non relasional (NoSQL). Tabel 3.5, Tabel 3.6, dan Tabel 3.7 adalah gambaran tabel atribut dan tipe data yang diterapkan pada MySQL.

Tabel 3.5 Perancangan Tabel Modul Mood

Mood	
id	Int (pk,auto increment)
judul	Varchar (250)
power	Text
harmonicity	Text

Tabel 3.6 Perancangan Tabel Modul Genre

Genre	
id	Int (pk,auto increment)
judul	Varchar (250)
asc	Text
ass	Text
asf	MediumBlob

Tabel 3.7 Perancangan Tabel Modul Tempo

Tempo	
id	Int (pk,auto increment)
judul	Varchar (250)
asc	Text
ass	Text
asf	MediumBlob

Untuk contoh isi data pada Tabel 3.5, Tabel 3.6, dan Tabel 3.7 diperlihatkan pada Gambar 3.10, Gambar 3.11, dan Gambar 3.12. Data yang disimpan berupa angka digital dalam bentuk *list* yang siap digunakan untuk proses pengolahan. Pada kolom judul merepresentasikan label dari suatu konten pada lagu. Contohnya *angry* untuk label pada *mood*, *classic* untuk label pada *genre*, dan *medium* untuk label pada tempo.

id	judul	power		harmonicity	
197	Angry	[0.0, 0.0, 0.0...	61K	[0.0, 0.7540582, 0....	51K
198	Angry	[0.0, 0.0, 0.0...	57K	[0.71663105, 0.4781...	51K
199	Angry	[0.0, 0.0, 0.0...	57K	[0.0, 0.98512477, 0...	51K
200	Angry	[0.0, 0.0, 0.0...	60K	[0.0, 0.35209343, 0...	51K
201	Angry	[0.0, 0.0, 0.0...	56K	[0.0, 0.95706576, 0...	51K
202	Angry	[0.0, 0.0, 0.0...	55K	[0.0, 0.9724374, 0....	51K
203	Angry	[0.0, 0.0, 0.0...	60K	[0.0, 0.9779616, 0....	51K
204	Angry	[0.0, 0.0, 0.0...	60K	[0.0, 0.90703976, 0...	51K
205	Angry	[0.0, 0.0, 0.0...	56K	[0.0, 0.9755724, 0....	51K
206	Angry	[0.0, 0.0, 0.0...	57K	[0.0, 0.98093385, 0...	51K

Gambar 3.10 Contoh Isi Data Pada Tabel Modul Mood

id	judul	asc		ass		asf	
1	Classic	[0.0, 0.0, 0....	55K	[0.0, 0...	50K	[1.0, 1.0, 1.0,...	1M
2	Classic	[0.0, 0.0, 0....	54K	[0.0, 0...	49K	[1.0, 1.0, 1.0,...	1M
3	Classic	[-2.057991, -...	54K	[1.9294...	51K	[0.994396, 0.99...	1M
4	Classic	[0.0, 0.0, 0....	53K	[0.0, 0...	48K	[1.0, 1.0, 1.0,...	1M
5	Classic	[-0.82308763,...	54K	[2.0427...	49K	[0.9960205, 0.9...	1M
6	Classic	[0.0, 3.04383...	54K	[0.0, 1...	50K	[1.0, 1.0, 1.0,...	1M
7	Classic	[0.0, 0.0, 0....	52K	[0.0, 0...	50K	[1.0, 1.0, 1.0,...	1M
8	Classic	[0.0, 0.0, 0....	53K	[0.0, 0...	50K	[1.0, 1.0, 1.0,...	1M
9	Classic	[0.0, 0.0, 0....	53K	[0.0, 0...	49K	[1.0, 1.0, 1.0,...	1M
10	Classic	[0.0, 0.0, 0....	53K	[0.0, 0...	49K	[1.0, 1.0, 1.0,...	1M
11	Classic	[0.0, 0.0, 0....	54K	[0.0, 0...	49K	[1.0, 1.0, 1.0,...	1M
12	Classic	[0.0, 0.0, 0....	53K	[0.0, 0...	49K	[1.0, 1.0, 1.0,...	1M

Gambar 3.11 Contoh Isi Data Pada Tabel Modul Genre

	id	judul	asc		ass		asf	
	1	Medium	[0.0, 0.0, 0.0, 0.0, 0.5938...	55K	[0.0, 0.0, 0.0,...	50K	[1.0, 1.0, 1....	1M
	2	Medium	[0.0, 0.0, 0.0, -0.9159667,...	53K	[0.0, 0.0, 0.0,...	49K	[1.0, 1.0, 1....	1M
	3	Medium	[0.0, 0.0, 0.0, 0.0, 0.0, 2...	54K	[0.0, 0.0, 0.0,...	48K	[1.0, 1.0, 1....	1M
	4	Medium	[0.0, 0.0, 0.0, 0.0, 0.4950...	55K	[0.0, 0.0, 0.0,...	48K	[1.0, 1.0, 1....	1M
	5	Medium	[0.0, 3.043834, -1.8201696,...	54K	[0.0, 0.0, 0.0,...	49K	[1.0, 1.0, 1....	1M
	6	Medium	[0.0, 0.0, 0.0, 0.0, 0.2301...	52K	[0.0, 0.0, 0.0,...	49K	[1.0, 1.0, 1....	1M
	7	Medium	[0.0, 0.0, 0.0, 0.0, -2.064...	53K	[0.0, 0.0, 0.0,...	49K	[1.0, 1.0, 1....	1M
	8	Medium	[0.0, 0.0, 0.0, 0.0, -1.141...	53K	[0.0, 0.0, 0.0,...	48K	[1.0, 1.0, 1....	1M
	9	Medium	[0.0, 0.0, 0.0, 0.0, -2.327...	53K	[0.0, 0.0, 0.0,...	49K	[1.0, 1.0, 1....	1M
	10	Medium	[0.0, 0.0, 0.0, 0.0, -1.107...	53K	[0.0, 0.0, 0.0,...	49K	[1.0, 1.0, 1....	1M

Gambar 3.12 Contoh Isi Data Pada Tabel Modul Tempo

3.2.2. Perancangan Antar Muka

Subbab ini menjelaskan bagaimana rancangan antarmuka yang akan berinteraksi secara langsung dengan pengguna pada saat tahap implementasi.

3.2.2.1. Perancangan Tampilan Awal Aplikasi Dibuka

Pada tampilan awal ketika pengguna membuka aplikasi adalah terdapat 3 tombol yaitu *signature*, *cover*song, dan detail lagu. Tombol-tombol ini adalah fungsionalitas dari aplikasi MusicMoo. Pada pengerjaan Tugas Akhir ini, bagian yang dikerjakan adalah modul *mood*, *genre* dan tempo di mana fungsionalitas sistem diimplementasikan di dalam tombol detail lagu. Perancangan tampilan dapat dilihat pada Gambar 3.13.

Gambar 3.13 Perancangan Tampilan Awal Aplikasi

3.2.2.2. Perancangan Tampilan Menu Dalam Tombol

Pada tampilan menu ketika memencet salah satu tombol. Tampilan menu di dalam ketika tombol akan terlihat sama, hanya yang membedakan adalah fungsionalitasnya. Perancangan tampilan menu digambarkan pada Gambar 3.14 di mana terdapat 3 tombol yaitu rekam, berhenti, dan kirim. Tombol rekam memiliki fungsi ketika ditekan akan melakukan perekaman lagu yang ingin diproses. Tombol berhenti memiliki fungsi ketika ditekan untuk menyelesaikan atau menghentikan proses perekaman. Lagu yang direkam tadi akan menjadi rekaman dengan format .wav. Terakhir tombol kirim memiliki fungsi melakukan *upload* lagu yang selesai direkam menuju server.

Gambar 3.14 Perancangan Tampilan Menu Dalam Button

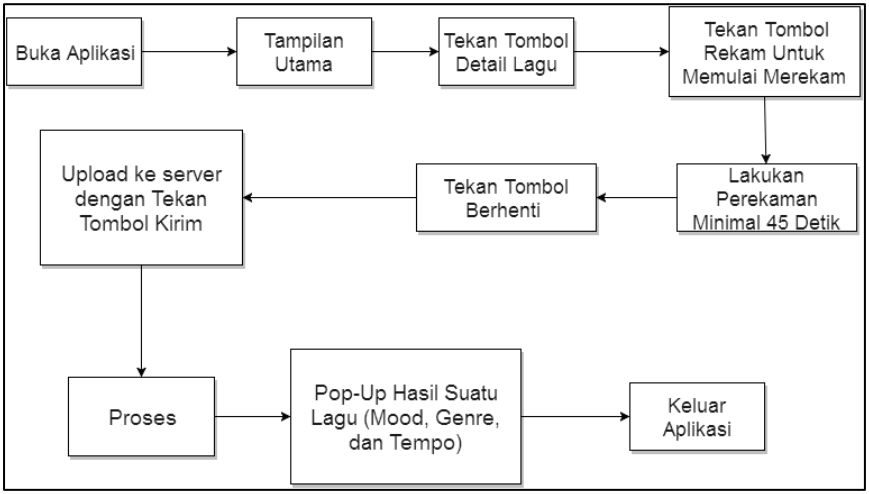
3.2.2.3. Perancangan Tampilan Hasil

Hasil yang ditampilkan pada aplikasi ini berbentuk *pop-up* notifikasi mengenai detail informasi lagu yang telah diproses. Perancangan tampilan hasil digambarkan pada Gambar 3.15. *Pop-up* dapat ditutup dengan cara menekan tulisan tutup pada pojok kanan bawah.

Gambar 3.15 Perancangan Tampilan Hasil

3.2.3. Perancangan Alur Proses Penggunaan Aplikasi

Pada tahap ini akan dijelaskan mengenai rancangan alur proses penggunaan aplikasi yang digunakan sebagai acuan dalam pembangunan aplikasi. Gambar 3.16 adalah penjelasan alur penggunaan aplikasi dalam bentuk *workflow*. Alur proses ini



Gambar 3.16 Workflow Alur Penggunaan Aplikasi

membantu memperlihatkan langkah demi langkah yang akan dilakukan pengguna sehingga terlihat jelas antara *input*, *proses*, dan *output.*

3.2.3.1. Penjelasan Alur Kerja Proses Penggunaan Aplikasi

Untuk alur proses, langkah-langkah yang dilakukan sama semua. Pertama lakukan perekeman suatu lagu minimal 45 detik. Ketika sudah selesai merekam, maka audio akan disimpan dalam ekstensi format .wav. Langkah berikutnya adalah *upload* rekaman lagu tadi ke server. Maka akan dilakukan proses-proses. Hasilnya berupa data *string.* Untuk modul *mood* yang dihasilkan adalah label *mood angry/happy/relaxed/sad*. Sedangkan untuk *genre*, label yang dihasilkan adalah *classic/electronic/jazz/rock.* Terakhir pada modul tempo label yang dihasilkan adalah *slow/ medium/fast*.

BAB IV
IMPLEMENTASI

Bab ini membahas implementasi dari analisis dan perancangan sistem yang telah dibahas pada Bab III. Namun dalam penerapannya, rancangan tersebut dapat mengalami perubahan minor sewaktu-waktu apabila dibutuhkan.

4.1. Lingkungan Implementasi

Dalam implementasinya, lingkungan yang digunakan sama seperti yang dituliskan pada rancangan, yakni menggunakan beberapa perangkat pendukung sebagai berikut.

4.1.1. Lingkungan Implementasi Perangkat Keras

Perangkat keras yang digunakan dalam implementasi pengembangan aplikasi ini adalah Komputer. Spesifikasi komputer yang digunakan adalah laptop dengan prosesor AMD E2-1800 APU *with* Radeon(tm) HD Graphics (2CPUs) dan RAM 4 GB.

4.1.2. Lingkungan Implementasi Perangkat Lunak

Penjelasan perangkat lunak yang digunakan dalam implementasi aplikasi ini adalah sebagai berikut:

- Microsoft Windows 7 Ultimate sebagai sistem operasi pada *notebook*.
- MySQL untuk mengimplementasikan rancangan basis data.

4.2. Implementasi Server

Subbab ini membahas tentang implementasi tampilan antarmuka yang telah dirancang dan dibahas pada Bab III. Selanjutnya akan dirinci dengan detail sebagai berikut.

4.2.1. Implementasi Konfigurasi Server dengan Flask

Kode Sumber 4.1 merupakan implementasi untuk melakukan konfigurasi server Flask. Dapat dilihat bahwa perlu dilakukan konfigurasi dulu yang terkait dengan nama *user*, *database*, *password,* serta *host.*

```
1.  from flask import Flask, render_template, session,
    redirect, url_for
2.  from flask.ext.mysql import MySQL
3.  import re
4.  import os
5.
6.  app = Flask(__name__)
7.  mysql = MySQL()
8.
9.  app.config['MYSQL_DATABASE_USER'] = 'MusicMoo2'
10. app.config['MYSQL_DATABASE_PASSWORD'] = '12345'
11. app.config['MYSQL_DATABASE_DB'] = 'MusicMoo'
12. app.config['MYSQL_DATABASE_HOST'] = '10.151.64.169'

13. mysql.init_app(app)
14.
15.
16. if __name__ == "__main__":
17.     app.run(host='0.0.0.0', port=5124, debug=True)
```

Kode Sumber 4.1 Konfigurasi Server Flask

4.2.2. Implementasi Mengubah Sinyal dari Time Domain menjadi Frekuensi Domain

Kode Sumber 4.2 adalah implementasi untuk mengubah sinyal dari *time domain* menjadi *frequency domain*. Tujuannya adalah untuk menganalisis sinyal mempunyai frekuensi maksimal berapa sebagai acuan level dekomposisi. *Input* yang diterima berupa sinyal dan *output*nya adalah hasil dari fungsi nilai dekomposisi level.

```
1.  def frekuensi(arr):
2.      npArr = np.array(arr)
3.      mean = np.mean(npArr)
4.      fin = npArr - mean
5.
6.      hasil = max(abs(np.fft.fft(fin.transpose(), axi
    s = 0) ) )
7.      fft = np.array(abs(np.fft.fft(fin.transpose(),
    axis=0 )))
```

```
8.      maxIndex = np.where(fft == hasil)[0][0]
9.
10.     fre = float(maxIndex) * 1024 / len(arr)
11.
12.     return nilaiDekomposisi(fre)
```

Kode Sumber 4.2 Mengubah Sinyal dari Time Domain Menjadi Frequency Domain

4.2.3. Implementasi Mencari Nilai Dekomposisi Terbaik

Kode Sumber 4.3 adalah implementasi untuk mencari nilai dekomposisi *wavelet* terbaik. Fungsi ini digunakan setelah sinyal dari *time* domain diubah menjadi frekuensi domain. *Input* yang diterima adalah hasil perhitungan pada variabel fre pada fungsi yang dijelaskan pada Kode Sumber 4.2. *Outputnya* adalah nilai level dekomposisi.

```
1.  def nilaiDekomposisi(fre):
2.      if (fre > 512):
3.          return 0
4.      elif(256 <= fre and fre <= 512):
5.          return 1
6.      elif(128 <= fre and fre <= 256):
7.          return 2
8.      elif(64 <= fre and fre <= 128):
9.          return 3
10.     elif(32 <= fre and fre <= 64):
11.         return 4
12.     elif(16 <= fre and fre <= 32):
13.         return 5
14.     elif(8 <= fre and fre <= 16):
15.         return 6
16.     elif(4 <= fre and fre <= 8):
17.         return 7
18.     elif(2 <= fre and fre <= 4):
19.         return 8
20.     elif(1 <= fre and fre <= 2):
21.         return 9
22.     elif(0.5 <= fre and fre <= 1):
```

```
        return 10
    elif(0.25 <= fre and fre <= 0.5):
        return 11
    elif(0.125 <= fre and fre <= 0.25):
        return 12
    elif(0.0625 <= fre and fre <= 0.125):
        return 13
```

Kode Sumber 4.3 Mencari Nilai Dekomposisi Terbaik

4.2.4. Implementasi *Discrete Wavelet Transform*

Kode Sumber 4.4 berikut ini adalah implementasi untuk melakukan *Discrete Wavelet Transform* (DWT). Fungsi ini digunakan untuk menghilangkan *noise* dari audio asli. *Input* yang diterima berupa sinyal lagu dan n level dekomposisi. *Outputnya* berupa sinyal yang sudah terwavelet dengan level dekomposisi terbaik.

```
import pywt

def waveletTransform(lagu, n):
    w = pywt.Wavelet('bior2.8')
    maks = pywt.dwt_max_level(data_len=len(lagu), f
ilter_len=w.dec_len)
    if(n > maks):
        n = maks
    hasil = pywt.wavedec(lagu, "bior2.8", level = n
)

    return hasil[0]
```

Kode Sumber 4.4 Melakukan Wavelet

4.2.1. Implementasi Reduksi Data

Kode Sumber 4.5 berikut ini adalah implementasi untuk melakukan penyamaan panjang fitur pada yang didapatkan. Tujuannya adalah menyeragamkan data agar dapat diklasifikasn menggunakan SVM. Fungsi ini dilakukan bersamaan dengan

menjalankan fungsi implementasi *Discrete Wavelet Transform* (DWT).

```
1.  panjangminimAP = 4498
2.  panjangminimAH = 4493
3.  panjangminimASC = 51
4.  panjangminimASS = 51
5.  panjangminimASF = 107904
6.
7.  #Contoh fitur yang dilakukan reduksi panjang#
8.  wvlaguAP18=dc.waveletTransform(AP18.AP,10)[:panjang
    minimAP]
9.  wvlaguAH18=dc.waveletTransform(AH18.AH,levelAH)[:pa
    njangminimAH]
10. wvlaguASC18=dc.waveletTransform(ASC18.ASC,levelASC)
    [:panjangminimASC]
11. wvlaguASS18=dc.waveletTransform(ASS18.ASS,levelASS)
    [:panjangminimASS]
12. wvlaguASF18=dc.waveletTransform(ASF18.ASF,levelASF)
    [:panjangminimASF]
13. wvlaguASP18=dc.waveletTransform(ASP18.ASP,levelASP)
    [:panjangminimASP]
```

Kode Sumber 4.5 Implementasi Reduksi Data

4.2.2. Implementasi Gabung Fitur

Kode Sumber 4.6 berikut adalah implementasi untuk melakukan penggabungan 2 atau 3 fitur ke dalam 1 *list*. *List* inilah yang nantinya digunakan untuk proses *training* yang digunakan untuk tahap klasifikasi menggunakan SVM. *Input* berupa fitur yang sudah dilakukan DWT beserta reduksi data. Untuk urutan tidak boleh terbalik, di mana *list* pada modul *mood* dengan panjang 0-4498 milik *Audio Power* dan sisanya *Audio Harmonicity*. Sedangkan *list* untuk modul *genre* dan tempo urutannya sebagai berikut: 51 panjang pertama untuk fitur *Audio Spectrum Centroid*, 51 panjang berikutnya untuk fitur *Audio Spectrum Spread*, dan sisanya untuk fitur *Audio Spectrum Flatness.*

```
1.  gabunganfitur18=list(merge(wvlaguAP18,wvlaguAH18))
    #membuat gabungan list berisi AP dan AH untuk modul
     mood
2.  gabunganfitur18=list(merge(wvlaguASC18,wvlaguASS18,
    wvlaguASF18)) #membuat gabungan list berisi ASC, AS
    S dan ASF untuk modul genre dan tempo
```

Kode Sumber 4.6 Implementasi Gabung Fitur

4.2.3. Implementasi SVM

Pada subbab ini dijelaskan implementasi proses klasifikasi untuk masing-masing modul.

4.2.3.1. Implementasi SVM Untuk Modul Mood

Kode sumber 4.7 merupakan implementasi untuk melakukan SVM pada modul *mood*. X adalah parameter untuk data *training* dan y adalah label dari masing-masing data *training*. Hasil prediksi adalah prediksi *mood* dari data *input* baru suatu lagu.

```
1.  def Mood(inputsinyal):
2.      X = np.array([
3.          #Isi List Data Training sebanyak 65x per la
    bel mood
4.      ])
5.
6.
7.      y = np.array([
8.          #Beri label pada masing masing list. Ada An
    gry, Happy, Relaxed, dan Sad sebanyal 65 per masing
    -masing label
9.          ])
10.
11.     from sklearn.svm import SVC
12.     clf = SVC()
13.     clf.fit(X,y)
14.     prediksi= clf.predict([inputsinyal]) //sinyal i
    nputan
15.     if prediksi == [1] : print 'Angry'
16.     elif prediksi== [2] : print 'Happy'
17.     elif prediksi == [3] : print 'Relaxed'
```

```
18.     else:  print 'Sad'
19.     return prediksi
```

Kode Sumber 4.7 SVM Pada Modul Mood

4.2.3.2. Implementasi SVM Untuk Modul Genre

Kode Sumber 4.8 merupakan implementasi untuk melakukan SVM pada modul *genre*. X adalah parameter untuk data *training* dan y adalah label dari masing-masing data *training.* Hasil prediksi adalah prediksi *genre* dari data *input* baru suatu lagu.

```
1.  def Genre (inputsinyal):
2.      X = np.array([
3.          #Isi List Data Training sebanyak 65x per ge
    nre lagu
4.      ])
5.
6.
7.      y = np.array([
8.          #Beri label pada masing masing list. Ada cl
    assic, elektronic, jazz, dan rock sebanyal 65 per m
    asing-masing label
9.          ])
10.
11.     from sklearn.svm import SVC
12.     clf = SVC()
13.     clf.fit(X,y)
14.     prediksi= clf.predict([inputsinyal]) //sinyal i
    nputan
15.     if prediksi == [1] : print 'Classic'
16.     elif prediksi== [2] : print ‘Electronic'
17.     elif prediksi== [3] : print ‘Jazz'
18.    else:  print 'Rock'
19.     return prediksi
```

Kode Sumber 4.8 SVM Pada Modul Genre

4.2.3.3. Implementasi SVM Untuk Modul Tempo

Kode Sumber 4.9 merupakan implementasi untuk melakukan SVM pada modul tempo. X adalah parameter untuk data *training* dan y adalah label dari masing-masing data *training*. Hasil prediksi adalah prediksi tempo dari data *input* baru suatu lagu.

```
1.  def Tempo (inputsinyal):
2.      X = np.array([
3.          #Isi List Data Training sebanyak 65x per la
    bel tempo
4.      ])
5.
6.
7.      y = np.array([
8.          #Beri label pada masing masing list. Ada sl
    ow, medium dan fast sebanyal 65 per masing-
    masing label
9.          ])
10.
11.     from sklearn.svm import SVC
12.     clf = SVC()
13.     clf.fit(X,y)
14.     prediksi= clf.predict([inputsinyal]) //sinyal i
    nputan
15.     if prediksi == [1] : print 'Fast'
16.     elif prediksi== [2] : print 'Medium'
17.    else:  print 'Slow'
18.     return prediksi
```

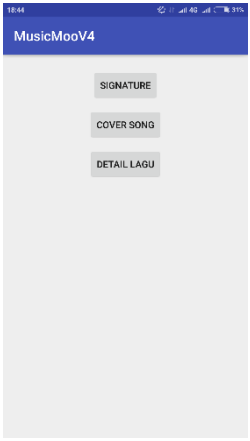
Kode Sumber 4.9 SVM Pada Modul Tempo

4.3. Implementasi Perangkat Bergerak

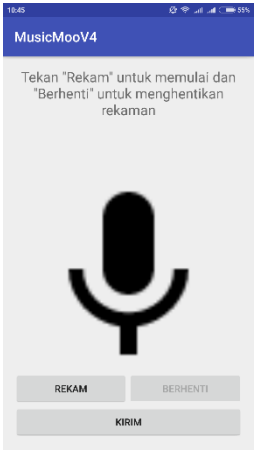
Pada subbab ini dijelaskan implementasi pada perangkat bergerak yang akan dibangun.

4.3.1. Implementasi Antar Muka

Berikut adalah tampilan implementasi antarmuka yang terlihat pada tampilan perangkat bergerak. Antarmuka dibangun sesuai yang dirancang pada subbab 3.2.2. Tampilan awal aplikasi dibuka dapat dilihat ada Gambar 4.1 sedangkan tampilan menu dalam suatu tombol dapat dilihat pada Gambar 4.2.



Gambar 4.1 Tampilan Awal Antarmuka Pada Perangkat Bergerak



Gambar 4.2 Tampilan Menu Di Dalam Tombol Pada Perangkat Bergerak

4.3.2. Implementasi Perekaman Audio

Kode Sumber 4.10 merupakan implementasi aplikasi Android untuk melakukan perekaman suara. Proses perekaman audio menggunakan *sample rate* 44100 Hz dan *encoding* PCM 16 Bit untuk mendapatkan hasil rekaman dengan kualitas terbaik.

```
1.  private void startRecording(){
2.      recorder = new AudioRecord(MediaRecorder.AudioS
    ource.MIC,
3.              RECORDER_SAMPLERATE, RECORDER_CHANNELS,
    RECORDER_AUDIO_ENCODING, bufferSize);
4.
5.      int i = recorder.getState();
6.      if(i==1)
7.          recorder.startRecording();
8.
9.      isRecording = true;
10.
11.     recordingThread = new Thread(new Runnable() {
12.
13.         @Override
14.         public void run() {
15.             writeAudioDataToFile();
16.         }
17.     },"AudioRecorder Thread");
18.
19.     recordingThread.start();
20. }
```

Kode Sumber 4.10 Implementasi Perekaman Audio

4.3.3. Implementasi Memberhentikan Perekaman Audio

Kode Sumber 4.11 merupakan implementasi aplikasi Android untuk memberhentikan proses perekaman suara.

```
1.  private void stopRecording(){
2.      if(null != recorder){
```

```
3.          isRecording = false;
4.
5.          int i = recorder.getState();
6.          if(i==1)
7.              recorder.stop();
8.          recorder.release();
9.
10.         recorder = null;
11.         recordingThread = null;
12.     }
13.
14.     copyWaveFile(getTempFilename(),getFilename());

15.     deleteTempFile();
16. }
```

Kode Sumber 4.11 Memberhentikan Perekaman Audio

4.3.4. Implementasi Penyimpanan Hasil Rekaman Audio

Kode Sumber 4.12 merupakan implementasi aplikasi Android untuk menyimpan hasil perekaman suara. Hasil yang disimpan adalah rekaman lagu dalam format wav.

```
1.  private String getFilename(){
2.      String filepath = Environment.getExternalStorag
    eDirectory().getPath();
3.      File file = new File(filepath,AUDIO_RECORDER_FO
    LDER);
4.
5.      if(!file.exists()){
6.          file.mkdirs();
7.      }
8.      selectedFilePath = file.getAbsolutePath() + "/"
     + "clip" + AUDIO_RECORDER_FILE_EXT_WAV;
9.  rn (file.getAbsolutePath() + "/" + "clip" + AUDIO_R
    ECORDER_FILE_EXT_WAV);
10. }
```

Kode Sumber 4.12 Penyimpanan Hasil Rekaman Audio

4.3.5. Implementasi Upload Lagu Ke Server

Kode Sumber ini merupakan implementasi aplikasi Android untuk meng-*upload* hasil perekaman suara ke server. Kode sumber dapat dilihat pada Lampiran Kode Sumber A.1 yang terletak pada lampiran.

4.4. Implementasi Ekstraktor MPEG-7

Pada subbab ini dijelaskan implementasi proses ekstraksi audio fitur berbasis MPEG-7.

4.4.1. Implementasi Mengimpor Library yang Digunakan

Kode Sumber 4.13 merupakan implementasi untuk mengimpor *library* yang digunakan yaitu *basex*, *basex-api*, *basex-xqj*-1.2.3, dan *MPEG7AudioEnc*. *Library* yang terdapat kata *basex* digunakan untuk melakukan *Xquery*. Sedangkan *MPEG7AudioEnc* digunakan untuk membuat metadata XML hasil ekstraksi MPEG-7.

```
1.  import org.basex.core.*;
2.  import org.basex.core.cmd.*;
3.  import org.basex.io.serial.*;
4.  import org.basex.query.*;
5.  import org.basex.query.iter.*;
6.  import org.basex.query.value.*;
7.  import org.basex.query.value.item.*;
8.
9.  import java.io.IOException;
```

Kode Sumber 4.13 Mengimpor Library yang Digunakan

4.4.2. Implementasi Ekstraktor basis MPEG-7

Kode Sumber 4.14 merupakan implementasi untuk ekstrasi audio fitur metadata berbasis MPEG-7. Tujuannya adalah untuk mengelolah fitur yang dimiliki suatu audio untuk memperkaya informasi. *Input* berupa direktori *path file* wav dan

output yg dihasilkan adalah metadata XML MPEG-7 yang berada di direktori *file* tempat audio disimpan.

```
1.  public class Descriptor{
2.      public static void main(String[] args){
3.          try {
4.              String sourceConfig = "C:\\Users\\Andre
    \\Documents\\NetBeansProjects\\MPEG-
    7_Extract\\config\\config.xml";
5.              InputStream inputStream = new FileInput
    Stream(sourceConfig);
6.              Reader reader = new InputStreamReader(i
    nputStream);
7.              Config config = ConfigXML.parse(reader)
    ;
8.              File folder = new File("D:\\Datasets TA
    \\");
9.              File[] listFile = folder.listFiles();
10.             for(int i = 0 ; i < listFile.length ; i
    ++) {
11.                 if(listFile[i].isFile()) {
12.                     AudioInputStream audioInputStre
    am = AudioSystem.getAudioInputStream(listFile[i]);

13.                     Document mpeg7 = MP7DocumentBui
    lder.encode(audioInputStream, config);
14.                     DOMSource domSource = new DOMSo
    urce(mpeg7);
15.                     StringWriter stringWriter = new
     StringWriter();
16.                     StreamResult result = new Strea
    mResult(stringWriter);
17.                     TransformerFactory transformerF
    actory = TransformerFactory.newInstance();
18.                     Transformer transformer = trans
    formerFactory.newTransformer();
19.                     transformer.transform(domSource
    , result);
20.                     String hasil = stringWriter.toS
    tring();
```

```
21.                     File file1 = new File("D:\\Data
    sets TA\\" + listFile[i].getName().split(".wav")[0]
     + ".xml");
22.                     FileWriter fileWriter = new Fil
    eWriter(file1, false);
23.                     fileWriter.write(hasil);
24.                     stringWriter.flush();
25.                     fileWriter.flush();
26.                     fileWriter.close();
27.                     System.out.println("i :" + i);

28.                 }
29.             }
30.         } catch (IOException e) {
31.             System.out.println(e);
32.         }catch (UnsupportedAudioFileException e){
33.             System.out.println(e);
34.         } catch (ParserConfigurationException e){
35.             System.out.println(e);
36.         } catch (TransformerException e){
37.             System.out.println("e");
38.         } catch (SAXException e){
39.             System.out.println("error baru");
40.         }
41.     }
42. }
```

Kode Sumber 4.14 Implementasi Ektraktor basis MPEG-7

4.5. Implementasi XQuery untuk Mengambil Fitur Audio

Pada subbab ini dijelaskan implementasi pengambilan fitur dari metadata XML yang dihasilkan oleh ekstraksi fitur berbasis MPEG-7.

4.5.1. Mengambil Nilai Audio Power

Kode Sumber 4.15 merupakan kode untuk mengimplementasikan mengambil nilai *Audio Power*. Kode sumber berikut adalah penerapan *query* pada data XML dengan

metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Power*.

```
1.      public  static String[] AudioPowerType(String p
    ath) throws BaseXException{
2.           String query =
3.                   "declare default element namespace
    \"urn:mpeg:mpeg7:schema:2001\";" +
4.                           "declare namespace mpeg7 =
    \"urn:mpeg:mpeg7:schema:2001\";" +
5.                           "declare namespace xsi = \"
    http://www.w3.org/2001/XMLSchema-instance\";" +
6.                           "for $x in doc(\""+path+"\"
    )/Mpeg7/Description/MultimediaContent/Audio/AudioDe
    scriptor\n return if($x/@xsi:type=\"AudioPowerType\
    ")then data($x/SeriesOfScalar/Raw) else \"\"";
7.            String hasil = new XQuery(query).execute(c
    ontext);
8.          String[] hasil1 = hasil.split("
9.   ");
10.          for(int i = 0 ; i<hasil1.length;i++){
11.              System.out.print(hasil1[i].trim().repla
    ce(" ",","));
12.              if(i != hasil1.length-1)
13.                 System.out.print(",");
14.          }
15.         return hasil1;
16. XQuery(query).execute(context));
17.      }
```

Kode Sumber 4.15 Mengambil Nilai Audio Power

4.5.2. Mengambil Nilai Audio Harmonicity

Kode Sumber 4.16 merupakan implementasi untuk mengambil nilai *Audio Harmonicity*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Harmonicity*.

```
1.  public static String[] AudioHarmonicityType(String
    path) throws BaseXException{
```

```
2.
3.      String query =
4.               "declare default element namespace \"ur
    n:mpeg:mpeg7:schema:2001\";" +
5.                      "declare namespace mpeg7 = \"ur
    n:mpeg:mpeg7:schema:2001\";" +
6.                      "declare namespace xsi = \"http
    ://www.w3.org/2001/XMLSchema-instance\";" +
7.                      "for $x in doc(\""+path+"\")/Mp
    eg7/Description/MultimediaContent/Audio/AudioDescri
    ptor\n return if($x/@xsi:type=\"AudioHarmonicityTyp
    e\")then data($x/HarmonicRatio/SeriesOfScalar/Raw)
    else \"\"";
8.
9.  String hasil = new XQuery(query).execute(context);

10.    String[] hasil1 = hasil.split("
11. ");
12.     for(int i = 0 ; i<hasil1.length;i++){
13.         System.out.print(hasil1[i].trim().replace("
     ",","));
14.         if(i != hasil1.length-1)
15.            System.out.print(",");
16.     }
17.    return hasil1;
18.     //System.out.println(new XQuery(query).execute(
    context));
19. }
```

Kode Sumber 4.16 Mengambil Nilai Audio Harmonicity

4.5.3. Mengambil Nilai Audio Spectrum Centroid

Kode Sumber 4.17 merupakan implementasi untuk mengambil nilai *Audio Spectrum Centroid*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Spectrum Centroid*.

```
1.  public static String[] AudioSpectrumCentroidType(St
    ring path) throws BaseXException{
2.      String query =
```

```
3.              "declare default element namespace \"ur
    n:mpeg:mpeg7:schema:2001\";" +
4.                      "declare namespace mpeg7 = \"ur
    n:mpeg:mpeg7:schema:2001\";" +
5.                      "declare namespace xsi = \"http
    ://www.w3.org/2001/XMLSchema-instance\";" +
6.                      "for $x in doc(\""+path+"\")/Mp
    eg7/Description/MultimediaContent/Audio/AudioDescri
    ptor\n return if($x/@xsi:type=\"AudioSpectrumCentro
    idType\")then data($x/SeriesOfScalar/Raw) else \"\"
    ";
7.   String hasil = new XQuery(query).execute(context);

8.     String[] hasil1 = hasil.split("
9.  ");
10.     for(int i = 0 ; i<hasil1.length;i++){
11.         System.out.print(hasil1[i].trim().replace("
     ",","));
12.         if(i != hasil1.length-1)
13.            System.out.print(",");
14.     }
15.    return hasil1;
16.     //System.out.println(new XQuery(query).execute(
    context));
17. }
```

Kode Sumber 4.17 Mengambil Nilai Audio SpectrumCentroid

4.5.4. Mengambil Nilai Audio Spectrum Spread

Kode Sumber 4.18 merupakan implementasi untuk mengambil nilai *Audio Spectrum Spread*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Spectrum Spread.*

```
1.     public static String[] AudioSpectrumSpreadType(S
    tring path) throws BaseXException{
2.
3.         String query =
```

```
                "declare default element namespace \
"urn:mpeg:mpeg7:schema:2001\";" +
                        "declare namespace mpeg7 = \
"urn:mpeg:mpeg7:schema:2001\";" +
                        "declare namespace xsi = \"h
ttp://www.w3.org/2001/XMLSchema-instance\";" +
                        "for $x in doc(\""+path+"\")
/Mpeg7/Description/MultimediaContent/Audio/AudioDes
criptor\n return if($x/@xsi:type=\"AudioSpectrumSpr
eadType\")then data($x/SeriesOfScalar/Raw) else \"\
"";

String hasil = new XQuery(query).execute(context);

      String[] hasil1 = hasil.split("
");
       for(int i = 0 ; i<hasil1.length;i++){
           System.out.print(hasil1[i].trim().replac
e(" ",","));
           if(i != hasil1.length-1)
              System.out.print(",");
       }
      return hasil1;
       //System.out.println(new XQuery(query).execu
te(context));
   }
```

Kode Sumber 4.18 Mengambil Nilai Audio Spectrum Spread

4.5.5. Mengambil Nilai Audio Spectrum Flatness

Kode Sumber 4.19 merupakan implementasi untuk mengambil nilai *Audio Spectrum Flatness*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Spectrum Flatness*.

```
    public  static String[] AudioSpectrumFlatnessTyp
e(String path) throws BaseXException{
        String query =
```

```
3.                "declare default element namespace \
   "urn:mpeg:mpeg7:schema:2001\";" +
4.                        "declare namespace mpeg7 = \
   "urn:mpeg:mpeg7:schema:2001\";" +
5.                        "declare namespace xsi = \"h
   ttp://www.w3.org/2001/XMLSchema-instance\";" +
6.                        "for $x in doc(\""+path+"\")
   /Mpeg7/Description/MultimediaContent/Audio/AudioDes
   criptor\n return if($x/@xsi:type=\"AudioSpectrumFla
   tnessType\")then data($x/SeriesOfVector/Raw) else \
   "\"";
7.  String hasil = new XQuery(query).execute(context);

8.        String[] hasil1 = hasil.split("
9.  ");
10.        for(int i = 0 ; i<hasil1.length;i++){
11.            System.out.print(hasil1[i].trim().replac
   e(" ",","));
12.            if(i != hasil1.length-1)
13.               System.out.print(",");
14.        }
15.       return hasil1;
16.    }
```

Kode Sumber 4.19 Mengambil Nilai Audio Spectrum Flatness

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada aplikasi yang dikembangkan. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsionalitas sistem yang telah dijabarkan pada Bab III dan terhadap tujuan dibuatnya aplikasi ini, yakni agar musik memiliki informasi yang lebih mendetail dan makin lengkap.

5.1. Lingkungan Pengujian

Lingkungan pengujian sistem pada pengerjaan Tugas Akhir ini dilakukan pada lingkungan dan alat kakas sebagai berikut:

Prosesor : Prosesor Intel® Core™ i3-CPU
RAM : 4 GB
Jenis *Device* : Personal Computer
Sistem Operasi : Ubuntu 16.04 LTS

Sedangkan perangkat bergerak yang digunakan dalam pengujian adalah sebagi berikut:

Prosesor : Mediatek MT6795 Helio X10
RAM : 2 GB
Jenis *Device* : Perangkat bergerak
Sistem Operasi : Android v5.1 Lollipop

5.2. Skenario Pengujian

Pada bagian ini akan dijelaskan tentang skenario pengujian yang dilakukan. Uji coba dilakukan untuk setiap modul yaitu *mood, genre*, dan tempo. Terdapat beberapa skenario saat ujicoba, dikarenakan hasil *output* mempengaruhi skenario yang digunakan:

1. Skenario 1: Melakukan uji coba lewat data langsung.
   - Melakukan ekstraksi fitur.
   - Mengambil fitur dengan *Xquery*.
   - Lakukan DWT dan *training data.*

- Uji coba dengan suatu *testing*
- Ekstrak lagu yang ingin dicoba, ambil fiturnya dengan *Xquery*.
- Lakukan DWT dan *training* data.
- Prediksi suatu data *input*.
- Hasil akan keluar sesuai dengan pembelajaran mesin.
- Hasil konstan tidak berubah-ubah.

2. Skenario 2: Melakukan uji coba lewat aplikasi perangkat bergerak.
   - Sebuah lagu, direkam melalui aplikasi perangkat bergerak minimal 45 detik, jika sudah tekan tombol *stop*.
   - *Upload* lagu dalam server.
   - Server akan melakukan ekstraksi fitur dan mengambil fitur yang akan diproses dengan *Xquery*.
   - Sistem menyimpan fitur yang dipakai dalam *database*.
   - Melakukan *processing* pada server Python.
   - Ambil data dari *database* lakukan DWT.
   - Lakukan *training* data.
   - Sistem akan memprediksi data baru (rekaman yang di-*upload* tadi).
   - Hasil akan keluar sesuai dengan pembelajaran mesin.
   - Hasil bisa berubah-rubah dikarenakan dipengaruhi oleh: *speaker* sumber suara/perekam suara dan *noise* suatu lingkungan.

Pada data akurasi Tugas Akhir ini, dilakukan uji coba menggunakan perangkat bergerak sesuai yang dijelaskan pada bab perancangan sistem. Tabel 5.1 memperlihatkan skenario pengujian pada tiap modul.

Tabel 5.1 Skenario Pengujian

Kode Pengujian	Skenario Pengujian
SP-UC1	Pengujian Modul *Mood*
SP-UC2	Pengujian Modul *Genre*
SP-UC3	Pengujian Modul Tempo

5.2.1. Pengujian Fungsionalitas

Pengujian fungsionalitas aplikasi dilakukan secara mandiri dengan melakukan skenario yang sama dengan rancangan alur proses aplikasi sebagai tolok ukur keberhasilan pengujian, dan mengacu pada kasus penggunaan yang sebelumnya telah dijelaskan pada Bab III. Pengujian pada kebutuhan fungsionalitas dapat dijabarkan pada subbab berikut.

5.2.1.1. Pengujian Modul *Mood*

Pemgujian pada modul *mood* ini dimulai dengan pengguna melakukan perekaman pada suatu lagu. Perekaman dilakukan melalui aplikasi perangkat bergerak. Rincian skenario pengujian pada kasus penggunaan dapat dilihat pada Tabel 5.2.

Tabel 5.2 Tabel Pengujian Modul Mood

ID	**SP-UC1**
Referensi Kasus Penggunaan	UC-1
Nama	Mengetahui *Mood* Sebuah Lagu.
Tujuan Pengujian	Pengguna mengetahui *mood* pada sebuah lagu.
Skenario	Mendeteksi *mood* sebuah lagu
Kondisi Awal	Pengguna membuka aplikasi MusicMoo.
Data Uji	Merekam suatu lagu minimal sepanjang 45 detik.
Langkah Pengujian	1. Pengguna membuka menekan tombol Detail Lagu 2. Pengguna melakukan perekaman sebuah lagu dengan memencet

	tombol rekam dengan durasi minimal 45 detik. 3. Setelah selesai, tekan tombol berhenti. 4. Lakukan proses upload pada lagu yang sudah direkam tadi dengan menekan tombol kirim. 5. Aplikasi perangkat bergerak akan melakukan proses. 6. Aplikasi akan menampilkan hasil berupa *pop-up*.
Hasil Yang Diharapkan	Tertampil detail lagu berupa label *mood*, *genre*, dan tempo.
Hasil Yang Didapatkan	Detail lagu *mood*, *genre*, dan tempo.
Hasil Pengujian	Berhasil.
Kondisi Akhir	Aktor melihat tampilan *pop-up* pada aplikasi perangkat bergerak berupa hasil label yang didapatkan.

5.2.1.2. Pengujian Modul *Genre*

Pengujian pada modul *genre* ini dimulai dengan pengguna melakukan perekaman pada suatu lagu. Perekaman dilakukan melalui aplikasi perangkat bergerak. Rincian skenario pengujian pada kasus penggunaan dapat dilihat pada Tabel 5.3.

Tabel 5.3 Tabel Pengujian Modul Genre

ID	SP-UC2
Referensi Kasus Penggunaan	UC-2
Nama	Mengetahui *Genre* Sebuah Lagu.
Tujuan Pengujian	Pengguna mengetahui *genre* pada sebuah lagu.
Skenario	Mendeteksi *genre* sebuah lagu

Kondisi Awal	Pengguna membuka aplikasi MusicMoo.
Data Uji	Merekam suatu lagu minimal sepanjang 45 detik.
Langkah Pengujian	1. Pengguna membuka menekan tombol Detail Lagu 2. Pengguna melakukan perekaman sebuah lagu dengan memencet tombol rekam dengan durasi minimal 45 detik. 3. Setelah selesai, tekan tombol berhenti. 4. Lakukan proses upload pada lagu yang sudah direkam tadi dengan menekan tombol kirim. 5. Aplikasi perangkat bergerak akan melakukan proses. 6. Aplikasi akan menampilkan hasil berupa *pop-up*.
Hasil Yang Diharapkan	Tertampil detail lagu berupa label *mood*, *genre*, dan tempo.
Hasil Yang Didapatkan	Detail lagu *mood*, *genre*, dan tempo.
Hasil Pengujian	Berhasil.
Kondisi Akhir	Aktor melihat tampilan *pop-up* pada aplikasi perangkat bergerak berupa hasil label yang didapatkan.

5.2.1.3. Pengujian Modul Tempo

Pengujian pada modul tempo ini dimulai dengan pengguna melakukan perekaman pada suatu lagu. Perekaman dilakukan

melalui aplikasi perangkat bergerak. Rincian skenario pengujian pada kasus penggunaan dapat dilihat pada Tabel 5.4.

Tabel 5.4 Tabel Pengujian Modul Mood

ID	SP-UC3
Referensi Kasus Penggunaan	UC-3
Nama	Mengetahui *Tempo* Sebuah Lagu.
Tujuan Pengujian	Pengguna mengetahui *genre* pada sebuah lagu.
Skenario	Mendeteksi *tempo* sebuah lagu
Kondisi Awal	Pengguna membuka aplikasi MusicMoo.
Data Uji	Merekam suatu lagu minimal sepanjang 45 detik.
Langkah Pengujian	1. Pengguna membuka menekan tombol Detail Lagu 2. Pengguna melakukan perekaman sebuah lagu dengan memencet tombol rekam dengan durasi minimal 45 detik. 3. Setelah selesai, tekan tombol berhenti. 4. Lakukan proses upload pada lagu yang sudah direkam tadi dengan menekan tombol kirim. 5. Aplikasi perangkat bergerak akan melakukan proses. 6. Aplikasi akan menampilkan hasil berupa *pop-up*.
Hasil Yang Diharapkan	Tertampil detail lagu berupa label *mood*, *genre*, dan tempo.
Hasil Yang Didapatkan	Detail lagu *mood*, *genre*, dan tempo.

Hasil Pengujian	Berhasil.
Kondisi Akhir	Aktor melihat tampilan *pop-up* pada aplikasi perangkat bergerak berupa hasil label yang didapatkan.

5.3. Akurasi Pengujian Fungsionalitas

Pada subbab ini akan diberikan hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang diberikan meliputi evaluasi pengujian kebutuhan fungsional sesuai dengan modul yang terdapat aplikasi.

Pada pengujian, tidak semua fitur dipakai. Dari tinjauan pustaka, maka dilakukan seleksi fitur sesuai definisi yang dijelaskan pada subbab 2.2 yang sesuai dengan masing-masing modul. Tabel 5.6 memperlihatkan fitur mana yang dipakai, mana yang tidak sesuai dengan definisi, dan mana fitur yang ketika dipakai hasilnya kurang baik.

Tabel 5.5 Pemilihan Tiap Fitur

Fitur	Modul		
	Mood	**Tempo**	**Genre**
Audio Power	V	**	**
Audio Waveform	X	X	X
Temporal Centroid	X	X	X
Log Attack Time	X	X	X
Audio Spectrum Centroid	X	V	V
Audio Harmonicity	V	**	**
Harmonic Spectral Centroid	X	X	X
Harmonic Spectral Deviation	X	X	X
Harmonic Spectral Spread	X	X	X
Harmonic Spectral Variaton	X	X	X
Audio Spectrum Spread	**	V	V

Audio Spectrum Projection	**	**	**
Audio Spectrum Basis	**	**	**
Audio Spectrum Flatness	**	V	V
Audio Spectrum Envelope	X	X	X
Audio Fundamental Frequency	X	X	X
Audio Signature	X	X	X

Keterangan simbol:
V = Sesuai dan dipakai.
X = Tidak dipakai karena tidak sesuai definisi.
** = Sesuai tapi ketika dilakukan uji coba hasil *output* jelek.

Akurasi dihitung menggunakan Persamaan 5.1 yang merupakan perhitungan dengan cara *Positive Preditive Value* (PPV) sebagai berikut:

$$PPV = \frac{TP}{TP + FP} * 100\% \tag{5.1}$$

Di mana *TP* adalah *True Positif* yaitu jumlah data yang diprediksi benar sesuai dengan labelnya dan *FP* adalah *False Positif* yaitu jumlah data yang diprediksi dikatakan benar namun kenyataannya adalah salah. *TP* + *FP* bisa juga dibilang jumlah data keseluruhan yang dipakai dalam pengujian.

Data yang dilakukan untuk uji coba pada masing-masing modul pada Tugas Akhir ini diperlihatkan pada Tabel 5.7.

Tabel 5.6 Rincian Jumlah Data Testing

Rincian Jumlah Data Testing			
Modul	**Label**	**Jumlah Data Testing**	**Total**
Mood	Angry	5	35
	Happy	10	
	Relaxed	10	
	Sad	10	
Genre	Classic	6	24
	Electronic	6	
	Jazz	6	
	Rock	6	
Tempo	Slow	10	30
	Medium	10	
	Fast	10	

Untuk detail lagu pada data *testing* masing-masing modul dapat dilihat pada Tabel A.1, Tabel A.2, dan Tabel A.3 yang terletak pada lampiran. Berikut penjelasan uji coba yang dilakukan pada subbab di bawah.

5.3.1. Hasil Percobaan Modul Mood

Berdasarkan definisi di atas, maka uji coba yang dilakukan untuk *mood* adalah memakai 2 fitur yaitu *Audio Power* dan *Audio Harmonicity*. Hasil yang diperoleh dilampirkan pada Tabel 5.8 sebagai berikut.

Tabel 5.7 Hasil Percobaan Modul Mood

Aktual	Pengujian			
	Angry	Happy	Relaxed	Sad
Angry	4	1	0	0
Happy	0	10	0	0
Relaxed	0	3	2	5
Sad	0	0	0	10

Dari data di atas, maka dilakukan penghitungan akurasi dengan Persamaan 5.1. Maka akurasi yang didapat dari uji coba modul *mood* adalah 75%.

5.3.2. Hasil Percobaan Modul Tempo

Berdasarkan definisi di atas, maka uji coba yang dilakukan untuk tempo adalah memakai 3 fitur yaitu *Audio Spectrum Centroid, Audio Spectrum Spread,* dan *Audio Spectrum Flatness.* Hasil yang diperoleh dilampirkan pada Tabel 5.9 sebagai berikut.

Tabel 5.8 Hasil Percobaan Modul Tempo

Aktual	Pengujian		
	Fast	Medium	Slow
Fast	5	5	0
Medium	0	10	0
Slow	0	1	9

Dari data di atas, maka dilakukan penghitungan akurasi dengan Persamaan 5.1. Maka akurasi yang didapat dari uji coba modul tempo adalah 80%.

5.3.3. Hasil Percobaan Modul Genre

Berdasarkan definisi di atas, maka uji coba yang dilakukan untuk *genre* adalah memakai 3 fitur yaitu *Audio Spectrum Centroid, Audio Spectrum Spread,* dan *Audio Spectrum Flatness.* Hasil yang diperoleh dilampirkan pada Tabel 5.10 sebagai berikut.

Tabel 5.9 Hasil Percobaan Genre

Aktual	Pengujian			
	Classic	Electronic	Jazz	Rock
Classic	6	0	0	0
Electronic	0	6	0	0
Jazz	1	1	4	0
Rock	0	1	0	5

Dari data di atas, maka dilakukan penghitungan akurasi dengan Persamaan 5.1. Maka akurasi yang didapat dari uji coba modul tempo adalah 87,5 %.

5.4. Evaluasi Pengujian

Pada subbab ini membahas hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang diberikan meliputi evaluasi pengujian kebutuhan fungsional beserta evaluasi berupa akurasi pada masing-masing modul.

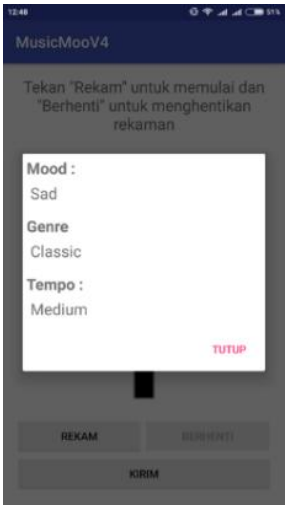
5.4.1. Evaluasi Pengujian Fungsionalitas

Rangkuman mengenai hasil pengujian fungsionalitas dapat dilihat pada Tabel 5.5. Berdasarkan data pada tabel tersebut, semua skenario pengujian berhasil dan program berjalan dengan baik. Sehingga bisa ditarik kesimpulan bahwa fungsionalitas dari aplikasi telah dapat bekerja sesuai dengan yang diharapkan.

Tabel 5.10 Rangkuman Hasil Pengujian

Kode Pengujian	Skenario Pengujian	Hasil
SP-UC1	Pengujian Modul *Mood*	Berhasil
SP-UC2	Pengujian Modul *Genre*	Berhasil
SP-UC3	Pengujian Modul Tempo	Berhasil

Gambar 5.1 adalah contoh hasil tampilan *output* berupa *pop-up* yang dilihat oleh pengguna. Pada pojok kanan bawah terdapat tulisan tutup untuk menutup *pop-up* yang dihasilkan.



Gambar 5.1 Contoh Tampilan Output Pada Aplikasi Perangkat Bergerak

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan Tugas Akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian perangkat lunak yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Fitur audio yang diekstrak dengan menggunakan basis MPEG-7 menghasilkan 17 fitur MPEG-7 *Low Level Descriptors*.
2. Pada modul *mood,* audio fitur yang mempengaruhi adalah *Audio Power* dan *Audio Harmonicity*.
3. Pada modul *genre,* audio fitur yang berpengaruh adalah fitur *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*.
4. Pada modul *tempo,* audio fitur yang berpengaruh adalah fitur *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*. Sama seperti dengan *genre*.
5. Tingkat akurasi untuk masing-masing modul sudah baik di mana *mood* 75%, *genre* 87,5%, dan tempo 80%.

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi dan pengujian yang telah dilakukan:

1. Peningkatan akurasi untuk masing-masing modul agar hasil semakin membaik

2. Analisis fitur-fitur MPEG-7 *Low Level Audio Descriptors* lainnya yang mungkin berpotensi mempengaruhi tiap-tiap modul juga.
3. Mengganti *database* relasional dengan *database* non relasional (NoSQL). Unuk pengerjaan masa datang disarankan memakai MongoDB.
4. Peningkatan tampilan antarmuka pada aplikasi perangkat bergerak.

DAFTAR PUSTAKA

[1] F. Wiering, "Can Humans Benefit from Music Information Retrieval?," in *Proceedings of the 4th International Conference on Adaptive Multimedia Retrieval: User, Context, and Feedback*, Berlin, Heidelberg, 2007, pp. 82–94.

[2] ISO/IEC (2001), "Information Technology — Multimedia Content Description Interface — Part 4: Audio," *FDIS 15938-4:2001*(*E*), June.

[3] M. Rocamora, P. Cancela, and A. Pardo, "Query by humming: Automatically building the database from music recordings," *Pattern Recognit. Lett.*, vol. 36, pp. 272–280, Jan. 2014.

[4] N. Chen, J. S. Downie, H. Xiao, and Y. Zhu, "Cochlear pitch class profile for cover song identification," *Appl. Acoust.*, vol. 99, pp. 92–96, Dec. 2015.

[5] G. Muhammad and M. Melhem, "Pathological voice detection and binary classification using MPEG-7 audio features," *Biomed. Signal Process. Control*, vol. 11, pp. 1–9, May 2014.

[6] "SoundHound is all things music." [Online]. Available: http://soundhound.com/soundhound. [Accessed: 14-Jan-2017].

[7] "Shazam - Music Discovery, Charts & Song Lyrics." [Online]. Available: https://www.shazam.com/. [Accessed: 14-Jan-2017].

[8] H.-G. Kim, N. Moreau, and T. Sikora, *MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval*. John Wiley & Sons, 2005.

[9] "Audio | MPEG." [Online]. Available: http://mpeg.chiariglione.org/standards/mpeg-7/audio. [Accessed: 13-Jan-2017].

[10] R. X. Gao and R. Yan, *Wavelets: Theory and Applications for Manufacturing*, 2011 edition. New York; London: Springer, 2010.

[11] B. T. Nugraha, R. Sarno, D. A. Asfani, T. Igasaki, and M. N. Munawar, "Classification of driver fatigue state based on EEG using Emotiv EPOC+," *J. Theor. Appl. Inf. Technol.*, vol. 86, no. 3, pp. 347–359, Apr. 2016.

[12] R. Sarno, M. N. Munawar, B. T. Nugraha, R. Sarno, M. N. Munawar, and B. T. Nugraha, "Real-Time Electroencephalography-Based Emotion Recognition System," *Int. Rev. Comput. Softw. IRECOS*, vol. 11, no. 5, pp. 456–465, May 2016.

[13] R. Sarno, B. T. Nugraha, M. N. Munawar, R. Sarno, B. T. Nugraha, and M. N. Munawar, "Real Time Fatigue-Driver Detection from Electroencephalography Using Emotiv EPOC+," *Int. Rev. Comput. Softw. IRECOS*, vol. 11, no. 3, pp. 214–223, Mar. 2016.

[14] M. N. Munawar, R. Sarno, D. A. Asfani, T. Igasaki, and B. T. Nugraha, "Significant preprocessing method in EEG-Based emotions classification," *J. Theor. Appl. Inf. Technol.*, vol. 87, no. 2, pp. 176–190, May 2016.

[15] D. R. Wijaya, R. Sarno, and E. Zulaika, "Information Quality Ratio as a novel metric for mother wavelet selection," *Chemom. Intell. Lab. Syst.*, vol. 160, pp. 59–71, Jan. 2017.

[16] M. Gruhne, R. Tous, J. Delgado, M. Doeller, and H. Kosch, "Introduction of an Mpeg-7 Query Format."

[17] B. Team, "The XML Database," 27-May-2015. [Online]. Available: http://basex.org/home/. [Accessed: 15-Jan-2017].

[18] M. Soleymani, M. N. Caro, E. M. Schmidt, C. Ya Sha, and Y.-H. Yang, "Emotion in Music Database - MediaEval 2013 - aka 1000 songs." [Online]. Available: http://cvml.unige.ch/databases/emoMusic/. [Accessed: 19-Dec-2016].

[19] M. Soleymani, M. N. Caro, E. M. Schmidt, C.-Y. Sha, and Y.-H. Yang, "1000 Songs for Emotional Analysis of Music," *Proc. ACM Int. Multimed. Conf. Exhib.*, vol. 6, no. 1, pp. 1–14, 2015.

[20] D. Hume, *Emotion and Moods*. Organizational behavior, 2012.
[21] "About the Music Genres List Site," *Music Genres List*. [Online]. Available: http://www.musicgenreslist.com/about-music-genre-site/. [Accessed: 15-Jan-2017].
[22] Y.-Y. Chang and Y.-C. Lin, "Music Tempo (Speed) Classification," 2005.
[23] "MixMeister BPM Analyzer," *Softonic*. [Online]. Available: https://mixmeister-bpm-analyzer.en.softonic.com/. [Accessed: 15-Jan-2017].
[24] M. Nardelli, G. Valenza, A. Greco, A. Lanata, and E. P. Scilingo, "Recognizing Emotions Induced by Affective Sounds through Heart Rate Variability," *IEEE Trans. Affect. Comput.*, vol. 6, no. 4, pp. 385–394, Oct. 2015.
[25] C.-H. Lin *et al.*, "SVM-Based Sound Classification Based on MPEG-7 Audio LLDs and Related Enhanced Features," in *Convergence and Hybrid Information Technology*, 2012, pp. 536–543.
[26] Z. W. Ras and A. Wieczorkowska, *Advances in Music Information Retrieval*, 1st ed. Springer Publishing Company, Incorporated, 2010.
[27] S. Li, H. Li, and L. Ma, "Music Genre Classification Based on MPEG-7 Audio Features," in *Proceedings of the Second International Conference on Internet Multimedia Computing and Service*, New York, NY, USA, 2010, pp. 185–188.

[Halaman ini sengaja dikosongkan]

LAMPIRAN A

LAMPIRAN KODE SUMBER

```
1.  private class PostDataTOServer extends AsyncTask<Vo
    id, Void, Void> {
2.          //Create hashmap Object to send parameters
    to web service
3.          HashMap<String, String> postDataParams;
4.          ProgressDialog progressDialog;
5.          TextView textView;
6.          @Override
7.          protected void onPreExecute() {
8.              super.onPreExecute();
9.
10.
11.             progressDialog = new ProgressDialog(Mai
    nActivity.this);
12.             progressDialog.setMessage("Please wait.
    ..");
13.             progressDialog.setCancelable(false);
14.             progressDialog.show();
15.
16.             textView = (TextView) findViewById(R.id
    .hasil);
17.         }
18.         @Override
19.         protected Void doInBackground(Void... arg0)
     {
20.             int serverResponseCode = 0;
21.
22.             HttpURLConnection connection;
23.             DataOutputStream dataOutputStream;
24.             String lineEnd = "\r\n";
25.             String twoHyphens = "--";
26.             String boundary = "*****";
27.
28.             int bytesRead,bytesAvailable,bufferSize
    ;
29.             byte[] buffer;
30.             int maxBufferSize = 1 * 1024 * 1024;
```

```
31.             File selectedFile = new File(selectedFi
    lePath);
32.
33.             String[] parts = selectedFilePath.split
    ("/");
34.             final String fileName = parts[parts.len
    gth-1];
35.
36.             try{
37.                 FileInputStream fileInputStream = n
    ew FileInputStream(selectedFile);
38.                 URL url = new URL(SERVER_URL);
39.                 connection = (HttpURLConnection) ur
    l.openConnection();
40.                 connection.setDoInput(true);//Allow
     Inputs
41.                 connection.setDoOutput(true);//Allo
    w Outputs
42.                 connection.setUseCaches(false);//Do
    n't use a cached Copy
43.                 connection.setRequestMethod("POST")
    ;
44.                 connection.setRequestProperty("Conn
    ection", "Keep-Alive");
45.                 connection.setRequestProperty("ENCT
    YPE", "multipart/form-data");
46.                 connection.setRequestProperty("Cont
    ent-Type", "multipart/form-
    data;boundary=" + boundary);
47.                 connection.setRequestProperty("uplo
    aded_file",selectedFilePath);
48.
49.                 //creating new dataoutputstream
50.                 dataOutputStream = new DataOutputSt
    ream(connection.getOutputStream());
51.
52.                 //writing bytes to data outputstrea
    m
53.                 dataOutputStream.writeBytes(twoHyph
    ens + boundary + lineEnd);
54.                 dataOutputStream.writeBytes("Conten
    t-Disposition: form-
    data; name=\"uploaded_file\";filename=\""
```

```
55.                         + selectedFilePath + "\"" +
     lineEnd);
56.
57.                 dataOutputStream.writeBytes(lineEnd
    );
58.
59.                 //returns no. of bytes present in f
    ileInputStream
60.                 bytesAvailable = fileInputStream.av
    ailable();
61.                 //selecting the buffer size as mini
    mum of available bytes or 1 MB
62.                 bufferSize = Math.min(bytesAvailabl
    e,maxBufferSize);
63.                 //setting the buffer as byte array
    of size of bufferSize
64.                 buffer = new byte[bufferSize];
65.
66.                 //reads bytes from FileInputStream(
    from 0th index of buffer to buffersize)
67.                 bytesRead = fileInputStream.read(bu
    ffer,0,bufferSize);
68.
69.                 //loop repeats till bytesRead = -
    1, i.e., no bytes are left to read
70.                 while (bytesRead > 0){
71.                     //write the bytes read from inp
    utstream
72.                     dataOutputStream.write(buffer,0
    ,bufferSize);
73.                     bytesAvailable = fileInputStrea
    m.available();
74.                     bufferSize = Math.min(bytesAvai
    lable,maxBufferSize);
75.                     bytesRead = fileInputStream.rea
    d(buffer,0,bufferSize);
76.                 }
77.
78.                 dataOutputStream.writeBytes(lineEnd
    );
79.                 dataOutputStream.writeBytes(twoHyph
    ens + boundary + twoHyphens + lineEnd);
80.
```

```
81.                 serverResponseCode = connection.get
    ResponseCode();
82.                 String serverResponseMessage = conn
    ection.getResponseMessage();
83.
84.                 Log.i(TAG, "Server Response is: " +
     serverResponseMessage + ": " + serverResponseCode)
    ;
85.
86.                 //response code of 200 indicates th
    e server status OK
87.
88.                 //closing the input and output stre
    ams
89.
90.                 String line;
91.                 BufferedReader br = new BufferedRea
    der(new InputStreamReader(connection.getInputStream
    ()));
92.                 //Log.d("Output",br.toString());
93.                 while ((line = br.readLine()) != nu
    ll) {
94.                     response += line;
95.                     Log.d("output lines", line);
96.                 }
97.
98.                 fileInputStream.close();
99.                 dataOutputStream.flush();
100.                        dataOutputStream.close();
101.
102.
103.
104.                } catch (FileNotFoundException e
    ) {
105.                    e.printStackTrace();
106.                    runOnUiThread(new Runnable()
     {
107.                        @Override
108.                        public void run() {
109.                            Toast.makeText(MainA
    ctivity.this,"File Not Found",Toast.LENGTH_SHORT).s
    how();
110.                        }
```

```
111.                         });
112.                     } catch (MalformedURLException e
    ) {
113.                         e.printStackTrace();
114.                         Toast.makeText(MainActivity.
    this, "URL error!", Toast.LENGTH_SHORT).show();
115.
116.                     } catch (IOException e) {
117.                         e.printStackTrace();
118.                         Toast.makeText(MainActivity.
    this, "Cannot Read/Write File!", Toast.LENGTH_SHORT
    ).show();
119.                     }
120.                     return null;
121.                 }
122.                 @Override
123.                 protected void onPostExecute(Void re
    sult) {
124.                     super.onPostExecute(result);
125.                     progressDialog.dismiss();
126.                     //Toast.makeText(MainActivity.th
    is,response,Toast.LENGTH_LONG).show();
127.                     textView.setText(response);
128.                 }
129.             }
130.         }
131.
```

Kode Sumber A.1 Upload Lagu ke Server

[Halaman ini sengaja dikosongkan]

LAMPIRAN B

LAMPIRAN TABEL

Tabel A.1 Hasil Percobaan Modul Mood

Data Test	Nilai Arousal	Nilai Valence	Label	Hasil Percobaan	Akurasi
962	7,4	3,4	Angry	Happy	FALSE
977	7,3	3,5	Angry	Angry	TRUE
980	5,8	4,3	Angry	Angry	TRUE
987	6,3	3,6	Angry	Angry	TRUE
995	4,7	2,4	Angry	Angry	TRUE
466	6,6	5,9	Happy	Happy	TRUE
467	5,7	6,8	Happy	Happy	TRUE
469	5,1	5,3	Happy	Happy	TRUE
472	6,5	5,2	Happy	Happy	TRUE
473	7,1	5,5	Happy	Happy	TRUE
475	5,6	5	Happy	Happy	TRUE
484	6,4	6,5	Happy	Happy	TRUE
486	6,2	5,7	Happy	Happy	TRUE
487	4,5	5,3	Happy	Happy	TRUE
489	4,6	5,2	Happy	Happy	TRUE
688	4,3	5	Relaxed	Relaxed	TRUE
819	4	5,7	Relaxed	Relaxed	TRUE
826	4	4,7	Relaxed	Sad	FALSE
829	2,3	4,5	Relaxed	Happy	FALSE
776	3,5	4,5	Relaxed	Happy	FALSE
779	2,8	5,3	Relaxed	Sad	FALSE
789	3,9	5,2	Relaxed	Happy	FALSE
791	2,7	4,6	Relaxed	Sad	FALSE
799	4,1	5,6	Relaxed	Sad	FALSE
811	4,1	4,9	Relaxed	Sad	FALSE
480	2,7	4,1	Sad	Sad	TRUE
482	1,9	3,5	Sad	Sad	TRUE

488	1,6	3,4	Sad	Sad	TRUE
494	3,7	4,1	Sad	Sad	TRUE
499	3,6	3,6	Sad	Sad	TRUE
637	2,2	3,6	Sad	Sad	TRUE
643	3,5	4,2	Sad	Sad	TRUE
646	3,1	3,8	Sad	Sad	TRUE
656	2,6	4,1	Sad	Sad	TRUE
660	2,1	2,8	Sad	Sad	TRUE
660	2,1	2,8	Sad	Sad	TRUE

Tabel A.2 Hasil Percobaan Modul Genre

Data Test	Label	Hasil Uji Coba	Akurasi
158	Classic	Classic	TRUE
159	Classic	Classic	TRUE
160	Classic	Classic	TRUE
164	Classic	Classic	TRUE
167	Classic	Classic	TRUE
168	Classic	Classic	TRUE
384	Electronic	Electronic	TRUE
430	Electronic	Electronic	TRUE
444	Electronic	Electronic	TRUE
490	Electronic	Electronic	TRUE
494	Electronic	Electronic	TRUE
498	Electronic	Electronic	TRUE
688	Jazz	Jazz	TRUE
643	Jazz	Jazz	TRUE
660	Jazz	Jazz	TRUE
740	Jazz	Jazz	TRUE
671	Jazz	Classic	FALSE

690	Jazz	Electronic	FALSE
962	Rock	Rock	TRUE
977	Rock	Rock	TRUE
980	Rock	Rock	TRUE
995	Rock	Rock	TRUE
991	Rock	Electronic	FALSE
999	Rock	Rock	TRUE

Tabel A.3 Hasil Percobaan Modul Tempo

Data Test	BPM	Label	Hasil Uji coba	Akurasi
714	158,24	Fast	Fast	TRUE
789	135,03	Fast	Fast	TRUE
979	146,61	Fast	Medium	FALSE
158	155,07	Fast	Medium	FALSE
187	159,84	Fast	Medium	FALSE
228	159,91	Fast	Medium	FALSE
444	137,3	Fast	Fast	TRUE
468	150,96	Fast	Medium	FALSE
494	159,8	Fast	Fast	TRUE
850	137,24	Fast	Fast	TRUE
408	120,35	Medium	Medium	TRUE
452	100,08	Medium	Medium	TRUE
459	129,03	Medium	Medium	TRUE
628	120,14	Medium	Medium	TRUE
662	107,37	Medium	Medium	TRUE
665	117,06	Medium	Medium	TRUE
674	107,97	Medium	Medium	TRUE
675	116,93	Medium	Medium	TRUE

687	134,91	Medium	Medium	TRUE
728	104,39	Medium	Medium	TRUE
231	97,05	Slow	Slow	TRUE
737	91,08	Slow	Slow	TRUE
221	98,09	Slow	Slow	TRUE
384	84,74	Slow	Slow	TRUE
430	97,33	Slow	Slow	TRUE
646	90,08	Slow	Slow	TRUE
775	99,73	Slow	Slow	TRUE
795	84,52	Slow	Slow	TRUE
848	89,99	Slow	Slow	TRUE
892	98,3	Slow	Medium	FALSE

BIODATA PENULIS

Johanes Andre Ridoean, lahir pada tanggal 6 Mei 1995 di Surabaya. Penulis menempuh pendidikan perguruan tinggi di Institut Teknologi Sepuluh Nopember Surabaya di Jurusan Teknik Informatika Fakultas Teknologi Informasi pada tahun 2013. Penulis memiliki pengalaman menjadi asisten dosen pada mata kuliah Teori Graf dan Manajemen Basis Data. Penulis terlibat aktif dalam organisasi kemahasiswaan dan kerohanian serta kegiatan kepanitiaan selama berkuliah, antara lain Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS pada tahun 2015-2016, staff Doa Pemerhati dan Konsumsi (DPK) di Pembinaan Kerohanian Mahasiswa Baru Kristen (PKMBK) ITS pada tahun 2015-2016, serta staff dana pada kegiatan Schematics tahun 2015. Dalam melakukan pengerjaan Tugas Akhir, penulis memiliki ketertarikan pada bidang Manajemen Informasi (MI). Untuk menghubungi penulis dapat melalui *email*: johanes.andre13@mhs.if.its.ac.id.